RFC 841

FIPS Pub 98

# SPECIFICATION FOR MESSAGE FORMAT FOR COMPUTER BASED MESSAGE SYSTEMS

27 January 1983

National Bureau of Standards

This RFC is FIPS 98. The purpose of distributing this document as an RFC is to make it easily accesible to the ARPA research community. This RFC does not specify a standard for the ARPA Internet.

TABLE OF CONTENTS


| | | Page |
|---|---|---|
| EXECUTIVE SUMMARY | | 5 |
| 1. | INTRODUCTION | 7 |
| 1.1 | Guide to Reading This Document | 7 |
| 1.2 | Vendor-Defined Extensions to the Specification | 8 |
| 1.3 | The Scope of the Message Format Specification | 8 |
| 1.4 | Issues Not Within the Scope of the Message Format Specification | 8 |
| 1.5 | Relationship to Other Efforts | 9 |
| 2. | A SIMPLE MODEL OF A CBMS ENVIRONMENT | 10 |
| 2.1 | Logical Model of a CBMS | 12 |
| 2.2 | Relationship to the ISO Reference Model for Open Systems Interconnection | 14 |
| 2.3 | Messages and Fields | 14 |
| 2.4 | Message Originators and Recipients | 15 |
| 3. | SEMANTICS | 17 |
| 3.1 | Semantics of Message Fields | 17 |
| 3.1.1 | Types of fields | 17 |
| 3.1.2 | Semantic Compliance Categories | 18 |
| 3.1.3 | Originator fields | 18 |
| 3.1.4 | Recipient fields | 19 |
| 3.1.5 | Date fields | 20 |
| 3.1.6 | Cross-reference fields | 21 |
| 3.1.7 | Message-handling fields | 22 |
| 3.1.8 | Message-content fields | 23 |
| 3.1.9 | Extensions | 23 |

3.2 Message Processing Functions 24
3.2.1 Message creation and posting 24
3.2.2 Message reissuing and forwarding 25
3.2.2.1 Redistribution 26
3.2.2.2 Assignment 28
3.2.3 Reply generation 28
3.2.4 Cross-referencing 29
3.2.4.1 Unique identifiers 29
3.2.4.2 Serial numbering 30
3.2.5 Life span functions 30
3.2.6 Requests for recipient processing 31
3.2.6.1 Message circulation 31
3.3 Multiple Occurrences and Ordering of Fields 31

4. SYNTAX 34

4.1 Introduction 34
4.1.1 Message structure 34
4.1.2 Data elements 35
4.1.2.1 Primitive data elements 36
4.1.2.2 Constructor data elements 36
4.1.3 Properties 36
4.1.3.1 Printing-names 37
4.1.3.2 Comments 37
4.1.4 Data compression and encryption 37
4.2 Overview of Syntax Encoding 37
4.2.1 Identifier Octets 38
4.2.2 Length code and Qualifier components 39
4.2.2.1 Length Codes 41
4.2.2.2 Qualifier 42
4.2.3 Property-List 44
4.2.4 Data Element Contents 44
4.3 Data Element Syntax 44
4.3.1 Data elements 45
4.3.1.1 Primitives 47
4.3.1.2 Constructors 49
4.3.1.3 Data Elements that Extend this Specification 52
4.3.2 Using data elements within message fields 53
4.3.3 Properties and associated elements 54
4.3.4 Encryption identifiers 54
4.3.5 Compression identifiers 54
4.3.6 Message types 55

SUMMARY OF APPENDIXES 56

APPENDIX A. FIELDS -- IMPLEMENTORS' MASTER REFERENCE 57

APPENDIX B. DATA ELEMENTS -- IMPLEMENTORS' MASTER REFERENCE 63

APPENDIX C. DATA ELEMENT IDENTIFIER OCTETS 71

APPENDIX D. SUMMARY OF MESSAGE FIELDS BY COMPLIANCE CATEGORY 72

D.1 REQUIRED Fields 72
D.2 BASIC Fields 72
D.3 OPTIONAL Fields 72

APPENDIX E. SUMMARY OF MESSAGE SEMANTICS BY FUNCTION 74

E.1 Circulation 74
E.2 Cross-Referencing 74
E.3 Life Spans 74
E.4 Delivery System 74
E.5 Miscellaneous Fields Used Generally 75
E.6 Reply Generation 75
E.7 Reissuing 75
E.8 Sending (Normal Transmission) 75

APPENDIX F. SUMMARY OF DATA ELEMENT SYNTAX 76

APPENDIX G. SUMMARY OF DATA ELEMENTS BY COMPLIANCE CATEGORY 78

G.1 BASIC Data Elements 78
G.2 OPTIONAL Data Elements 78

APPENDIX H. EXAMPLES 80

H.1 Primitive Data Elements 80
H.2 Constructor Data Elements 82
H.3 Data Elements that Extend this Specification 87
H.4 Fields 88
H.5 Messages 90
H.6 Unknown Lengths 94
H.7 Message Encoding Using Vendor-Defined Fields 97
H.7.1 Example of a JANAP-128 Message 97
H.7.2 Encoding of Example using the FIPS Message Format 97
H.7.3 Field Mappings of JANAP-128 to FIPS Format 101
H.7.4 Vendor-Defined Fields 101

REFERENCES 103

INDEX 105

# LIST OF FIGURES


| | | |
|---|---|---|
| FIG. 1. | LOGICAL MODEL OF A COMPUTER-BASED MESSAGE SYSTEM | 12 |
| FIG. 2. | MESSAGE FORWARDING AND REDISTRIBUTION | 27 |
| FIG. 3. | EXAMPLE OF MESSAGE CIRCULATION | 32 |
| FIG. 4. | STRUCTURE OF IDENTIFIER OCTETS | 39 |
| FIG. 5. | ENCODING MECHANISM FOR QUALIFIERS AND LENGTH CODES | 40 |
| FIG. 6. | REPRESENTATION OF LENGTH CODES | 42 |
| FIG. 7. | EXAMPLES OF QUALIFIER VALUES | 43 |

# LIST OF TABLES


| | | |
|---|---|---|
| TABLE 1. | FIELDS USED IN MESSAGE PROCESSING FUNCTIONS | 24 |
| TABLE 2. | HIGH-ORDER BITS IN THE IDENTIFIER OCTET | 39 |

Federal Information
Processing Standards Publication 98
27 January 1983
Announcing the Standard for

# MESSAGE FORMAT
# FOR
# COMPUTER BASED MESSAGE SYSTEMS

Federal Information Processing Standards Publications are issued by the National Bureau of Standards pursuant to section 111(f)(2) of the Federal Property and Administrative Services Act of 1949, as amended, Public Law 89-306 (79 Stat. 1127), Executive Order 11717 (38 FR 12315, dated May 11, 1973), and Part 6 of Title 15 Code of Federal Regulations (CFR).

Name of Standard. Message Format for Computer Based Message Systems (FIPS PUB 98).

Category of Standard. Software Standard; Interchange Codes, Media and Data Files.

Explanation. This standard separates information so that a Computer Based Message System can locate and operate on that information (which is found in the fields of messages). This is the first of a family of standards which will ensure information interchange among Computer Based Message Systems.

Approving Authority. Secretary of Commerce

Maintenance Agency. Department of Commerce, National Bureau of Standards (Institute for Computer Sciences and Technology).

Cross Index. Not Applicable.

Related Documents.

a. American National Standard Code for Information Interchange (ASCII), X3.4-1977,FIPS PUBS 1-1.

b. American National Standard Code Extension Techniques for Use with the 7-bit Coded Character Set of American National Standard Code (ASCII) for Information Interchange, X3.41-1974, FIPS PUB 35.

c. National Bureau of Standards. Calendar Date. Federal Information Processing Standards Publication 4, U.S.

Department of Commerce / National Bureau of Standards, November, 1968.

d. National Bureau of Standards. Data Encryption Standard. Federal Information Processing Standards Publication 46, U.S. Department of Commerce/National Bureau of Standards, January, 1977.

e. National Bureau of Standards. Representation of Local Time of the Day for Information Interchange. Federal Information Processing Standards Publication 58, U.S. Department of Commerce / National Bureau of Standards, February 1979.

f. National Bureau of Standards. Representation of Universal Time, Local Time Differentials, and United States Time Zone References for Information Interchange. Federal Information Processing Standards Publication 59, U.S. Department of Commerce / National Bureau of Standards, February, 1979.

Applicability. This message format standard applies to Federal departments and agencies in their acquisition and use of computer-based message systems (CBMS) and services in networked systems, except for certain single-processor systems. Specifically, the standard does not apply to a CBMS if it is a stand-alone system which is not interconnected with any other CBMS: nevertheless, conformance with the standard is recommended under these circumstances particularly if there is a possibility that use of another central processing unit, or interconnection with another system, will be required in the future. Where a new CBMS node is incorporated into an existing network, the standard applies at the interface between CBMS's. In this instance, previously existing nodes may accommodate the standard either through retrofit or by the use of a translator. In addition, networks that are established strictly for the purpose of supporting research in computer science or communications are exempt from complying with this standard.

Subcommittee TC97/SC16 of the International Organization for Standardization (ISO) has developed a reference model for describing communications between "open" systems. (ISO/TC97/SC16 DIS7498) This model is known as the ISO Reference Model for Open Systems Interconnection (OSI). It divides communications protocols into seven layers, ranging from physical interconnection at the lowest layer to data exchange by applications programs at the top.

The NBS message format deals with data used by an application within a system; it is not a protocol. Messages defined by the

NBS message format would be manipulated by a layer 7 (Application) protocol.

A message as referenced by the NBS message format is a unit of communication from an originator to a recipient, exclusive of any message heading or control information (often referred to as a message envelope). An originator and recipient are typically people but may be roles or processes. A role identifies a function within an organization as opposed to an individual who performs that function. A process refers to a computer process that might originate or receive a message.

Special Information. Certain characteristics distinguish a CBMS from other systems for sending messages. Originators and recipients may be terminal users or processes (discrete software). A system in which the originator addresses a particular terminal device rather than a particular recipient is not considered to be a CBMS. The recipient's system need not be available when the originator sends a message. The message can be stored in the originator's system or at an intermediate node in the network until the recipient's system becomes available. In addition, a CBMS offers both message creation and message processing facilities as part of the system. A CBMS offers text editing facilities to assist the user in the preparation of a message. The recipient CBMS stores the message until the recipient chooses to read it. Message systems which do not provide these minimum functions are not considered CBMS's.

The intent of the message format standard is to allow users of different computer based message systems to send messages to each other. The standard does not make demands on the message transfer system except that it transports messages transparently. The standard makes some simple demands on the CBMS. The CBMS must recognize fields within the message, process fields in predetermined ways, create messages in the correct form, and recognize and create data elements of messages in the correct format. The standard does not dictate or constrain the services that the CBMS provides for users, or the way that messages are stored, represented, manipulated, or presented to the user by the CBMS.

The standard does constrain the format of the message at the interface between systems. This guarantees that, whatever the source of the message, it arrives at the receiving system in the standard format. The message format standard separates information into fields so that the CBMS can locate and operate on that information. The message is converted from the format used within the originator's CBMS to the standard format (if different) on leaving the originator's CBMS. The message is converted from the standard format to the format used within the recipient's CBMS (if different) on entering the recipient's CBMS.

Specifications. Federal Information Processing Standard (FIPS), Message Format for Computer Based Message Systems (affixed).

Qualifications. None

Implementation Schedule. All applicable equipment or services ordered on or after 24 months from the date of issuance of this FIPS PUB, and all CBMS development initiated inhouse on or after 12 months from the date of issuance of this FIPS PUB must be in conformance with this standard unless a waiver has been obtained in accordance with the procedure described below. An exception to this standard is made when procurement actions are into the solicitation phase on the date of issuance of this FIPS PUB.

Waivers. Heads of agencies may request that the requirements of this standard be waived in instances where it can be clearly demonstrated that there are appreciable performance or cost advantages to be gained and that the overall interests of the Federal Government are best served by granting the requested waiver. Such waiver requests will be reviewed by and are subject to the approval of the Secretary of Commerce. The waiver request must address the criteria stated above as the justification for the waiver.

Forty-five days should be allowed for review and response by the Secretary of Commerce. Waiver requests shall be submitted to the Secretary of Commerce, Washington, D.C. 20230, and labeled as a Request for a Waiver to a Federal Information Processing Standard. No agency shall take any action to deviate from the standard prior to the receipt of a waiver approval from the Secretary of Commerce. No agency shall begin any process of implementation or acquisition of non-conforming equipment unless it has already obtained such approval.

Where to Obtain Copies. Either paper or microfiche copies of this Federal Information Processing Standard, including technical specifications, may be purchased from the National Technical Information Service (NTIS) by ordering Federal Information Processing Standard Publication (FIPS-PUB-98), Message Format for Computer Based Message Systems. Ordering information, including prices and delivery alternatives, may be obtained by contacting the National Technical Information Service (NTIS), U. S. Department of Commerce, Springfield, Virginia 22161, telephone number (703) 487-4650. Payment may be made by check, money order, purchase order, credit card, or deposit account.

# EXECUTIVE SUMMARY

The message format specification addresses the problem of exchanging messages between different computer-based message systems (CBMSs). This interchange problem can be addressed on several levels. One level specifies the physical interconnections, another specifies how information travels between CBMSs, another specifies form and meaning of messages being interchanged. The highest level specifies operations on a message. Each of these levels would be covered by a different standard.

This message format specification addresses only the issues of form and meaning of messages at the points in time when they are sent from one CBMS and received by another. Messages are composed of fields, containing different classes of information. These fields contain information about the message originator, message recipient, subject matter, precedence and security, and references to previous messages, as well as the text of the message. Standard formats (syntax) for messages provide a basis for the contents of messages generated by one CBMS to be processed by another CBMS. Standard meanings (semantics) for the components of a message facilitate standard interpretation of a message, so that everyone receiving a message gets the meaning intended by its sender.

Each CBMS that implements this message format specification will be compatible with any other CBMS that implements the specification, provided that the use of optional fields and data elements is negotiated in advance. This ensures that the contents of a message posted by one CBMS can be received and interpreted by a different CBMS.

This message format specification has been developed as a result of examining CBMSs currently in use in commercial and research environments. Three major design perspectives helped shape the message format specification.

- Viability. The message format specification uses concepts that already work. It has been designed with implementation concerns in mind.

- Compatibility. The message format specification contains concepts from existing CBMSs. For this reason, many CBMS would already contain functions and components similar to those required to implement the message format specification.

- Extensibility. This message format specification defines a broad range of message content components and requires only an elementary subset of them. This means that even a very simple CBMS can implement the message format specification. The message format specification contains a rich set of optional components and, in addition, mechanisms for user extensions and future extensions to the message format specification.

The message format specification defines the form and meaning of message contents and their components as they pass from one CBMS to another through a message transfer system. The message format specification does not address any of the following major issues.

- Functions or services provided to a user by a CBMS.
  For example, the message format specification assumes that every CBMS allows a user to send and receive messages. It does not specify any of the details of how a send function or a message-reading function might work or how it might appear to the user. That is, the message format specification neither limits nor mandates functions.

- Storage or format of message contents in a CBMS.
  The message format specification defines the form and contents of messages when they are transferred between systems. A CBMS may or may not choose to use the same format for internal storage.

- Message transfer system protocols.
  The message format specification does not specify how a message travels between CBMSs. It does specify the form of its contents as it leaves and arrives, assuming only that the message is moved transparently by the transfer system.

- Message envelopes.
  While a message is traveling between CBMSs, it is enclosed in a message envelope. Message envelopes contain all the information about a message that a message transfer system needs to know. The message format specification does not define the format or content of a message envelope.

- How message originators and recipients are identified.
  The message format specification does not provide a representation scheme for the names or addresses of message originators and recipients as they are known to a CBMS.

# 1. INTRODUCTION

A computer-based message system (CBMS) allows communication between "entities" (usually people) using computers. Computers serve both to mediate the actual communications between systems and to provide users with facilities for creating and reading the messages.

CBMSs have been developing for over ten years. More recently, CBMSs have been one of the bases in industry for the introduction of office automation. A growing number of organizations use either their own or a commercially available CBMS. The design and complexity of these systems vary widely. This message format specification provides a basis for interaction between different CBMSs by defining the format of messages passed between them.

## 1.1 Guide to Reading This Document

The method of presenting the material in this specification is to combine the technical specification with tutorial information. This approach has been taken to place the specification in context and improve its readability.

The core of the technical information in the document is in Section 2, "A Simple Model of a CBMS Environment"; Section 3.1, "Semantics of Message Fields"; Section 4.2, "Overview of Syntax Encoding"; and Section 4.3, "Data Element Syntax". Appendixes A and B consolidate the technical information. These appendices are designed for ease of reference and should be read in conjunction with the body of the report for a complete understanding of the message format presented in the specification.

Section 2 presents a simple model of operation of a CBMS. Section 3 discusses the components of messages and their meaning (semantics), including discussions of the recommended relationship between message components and CBMS user functions. (See Section 3.2.) Section 4 presents details of the form (syntax) required for components of a message.

Appendix D summarizes the components of messages according to whether they are required or optional for CBMSs implementing the message format specification. Appendix E organizes the message components according to the functional class of the components. Appendix F provides an overview of the syntactic elements defined by this message format specification; Appendix G

summarizes those elements according to whether they are required or optional for a CBMS implementing the message format specification. Examples of each syntactic element appear in Appendix H, displaying syntax and describing the associated semantics.

## 1.2 Vendor-Defined Extensions to the Specification

This specification provides the capability of extending the range of functionality by the use of vendor-defined qualifiers and vendor-defined data elements. Any vendor who uses this capability to provide services which are essentially equivalent to those already designated as required, basic, or optional does not comply with the specification.

## 1.3 The Scope of the Message Format Specification

The purpose of this message format specification is to present the semantics and syntax to be used for messages being exchanged between CBMSs. Specifically, it defines the following:

- The meaning and form of standard fields to be used in messages.
- Which fields must be present in all messages.
- Which fields complying CBMSs must be able to process.
- How messages, fields, and the data contained in fields are represented.

## 1.4 Issues Not Within the Scope of the Message Format Specification

The message format specification does not address the following issues, some of which are being covered by other NBS standards development programs at the Institute for Computer Sciences and Technology (ICST). (See [BlaR-80] for a description of the ICST network protocols program.)

- The nature of a message transfer system, except to state the assumption that it transfers messages transparently.

- The form or nature of the protocols used to transfer messages (posting, relay, and delivery protocols).
- The content and representation of message envelopes.
- Representations for unique identifiers (in particular, message identifiers).
- Network and internetwork addressing.
- Representations for identities of message originators and recipients.
- Certain message processing functions that CBMSs provide for users, e.g., those concerned with the creation and editing of text.
- Presentation of messages to users.
- Representations for multi-media objects.
- Data representation for messages within CBMSs.
- Data sharing or any storage management within CBMSs.
- Representations for fixed or floating point numbers.

## 1.5 Relationship to Other Efforts

The message format specification is based on several documents and the current state of many CBMSs available both in industry and the research community. These documents include the standardization efforts in the ARPANet [CroD-77, PosJ-79] and the CCITT, proposed ISO and ANSI header format standards [TasG-80, ISOD-79], the work of IFIPS Working Group 6.5, and various papers about the general nature of mail systems, addressing, and mail delivery. (See [FeiE-79] for references.

## 2. A SIMPLE MODEL OF A CBMS ENVIRONMENT

In order to provide a framework for presenting the message format specification, this section describes a simple functional model for a CBMS. The model provides a high-level description of both user facilities and system architecture. Discussions of messages, message originators, and message recipients serve to further clarify the nature of a CBMS.

A CBMS permits the transfer of a message from an originator to a recipient. "Originator" and "recipient" are used in their normal English senses. (See Section 2.4.) A message (in its most abstract definition) is simply a unit of communication from an originator to a recipient. A CBMS offers several classes of functions to its users:

- o Message Creation: The facilities used by a message originator to create messages and specify to whom they are to be sent.

- o Message Transfer: The facilities used to convey a message to its recipient(s).

- o Recipient Processing: The facilities used by a message recipient to process messages that have arrived.

These classes of functions are presented in more detail in Section 3.2.

CBMSs differ from other office automation/communications systems in a number of ways.

- o Unlike other types of electronic communications, CBMS messages are sent to particular individuals, not to stations or telephone sets. If a recipient moves to a different location, messages sent to that recipient are delivered to the recipient at the new location.

- o Transmission of CBMS messages is asynchronous. The recipient's system need not be available when the message leaves the originator's system. That is, CBMS message transfer facilities are store-and-forward.

- o CBMS messages can contain a wide variety of data. They are not constrained to any single kind of communication. CBMS messages are often simple memoranda but are not restricted to text. A CBMS message may contain any kind

of data that an originator wishes to send to a recipient. By contrast, Teletex systems and communicating word processors handle the transfer of final form documents; compatible communicating word processors can exchange documents in editable form; Telex and TWX deal in unformatted text.

- CBMSs offer message creation facilities as an important part of the system. CBMSs assist users in the preparation of messages by having text editing facilities available and allowing users to include data stored on-line in messages. Some CBMSs also interface to other office automation facilities, such as formatters and spelling correctors. This is not true of Telex, TWX, or similar services.

- CBMSs offer recipient processing facilities as an important part of the system. This is not true of most other forms of electronic communications. For example, Telex and TWX systems simply print messages on paper when they are received, without retaining a copy in the system. (Teletex systems are similar to Telex systems, but some can retain a copy of the document in local storage.) Communicating word processors might notify their operators that a document has been received and is stored on-line, but they offer little in the way of other recipient processing facilities. Most CBMSs offer at least the following recipient processing facilities:

  . The ability to retain a copy of a message on-line after it has been read.

  . The ability to examine or delete stored messages individually.

  . The ability to organize messages using some form of electronic "file folder."

  . The ability to determine if a message is recent (has arrived since the last time the recipient used the CBMS) or unseen (has never been examined by the recipient).

  . The ability to summarize stored messages. A summary usually includes information such as whether the message is recent or unseen, when it was received, its length, who it is from, and its subject.

  . The ability to retrieve a stored message based upon

one or more of its attributves (for example, when the message was received, whether or not it has been seen or deleted, and the values contained in its fields).

- A forward facility that allows users to include all or part of a message in a new outgoing message.

- A reply facility that allows users to answer messages without having to enter a new list of recipients.

## 2.1 Logical Model of a CBMS

CBMS facilities for message creation, transfer, and recipient processing are reflected in a logical model of a CBMS developed by IFIP Working Group 6.5. (An essentially identical model is being used by CCITT Study Group VII, Question 5, regarding Message Handling Systems [CCIT-82].) The model consists of a Message Transfer System and a number of User Agents. (See Figure 1.)

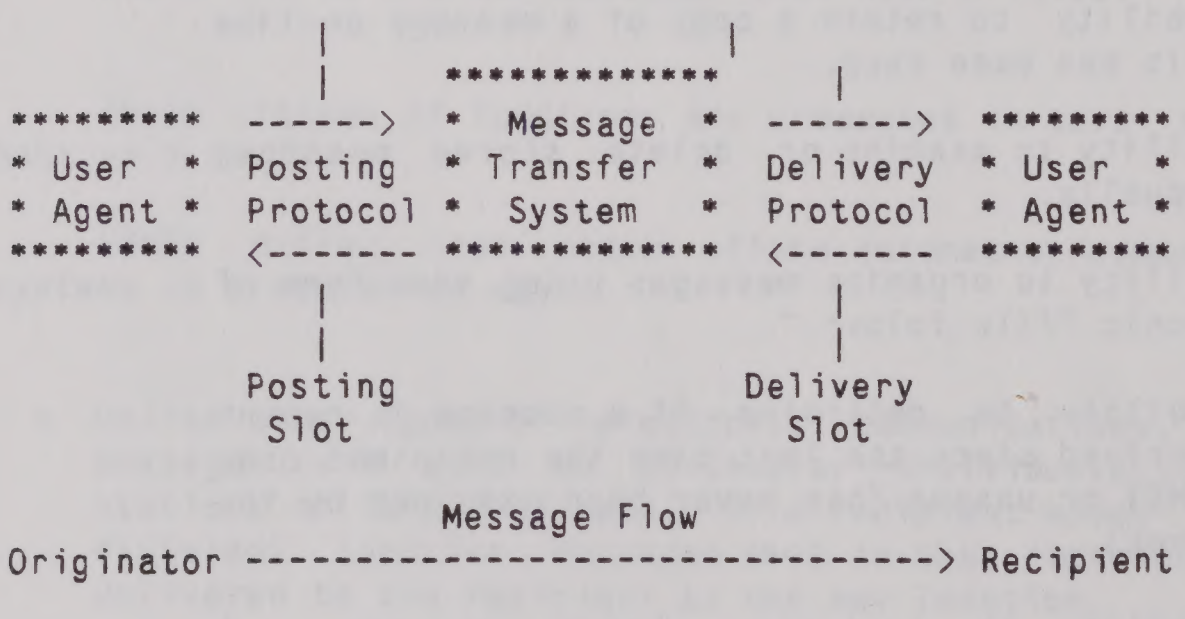



FIG. 1. LOGICAL MODEL OF A COMPUTER-BASED MESSAGE SYSTEM

A User Agent (UA) is a functional entity that acts on behalf of a user, assisting with creating and processing messages and communicating with the Message Transfer System.

The Message Transfer System(MTS) is an entity that accepts a

message from its originator's User Agent and ultimately passes it to each of its recipients' User Agents. The Message Transfer System may perform routing and storage functions (among others) in order to accomplish its task.

Transferring a message from an originator's User Agent to the Message Transfer System is called Posting; the originator's User Agent and Message Transfer System engage in a Posting Protocol in order to accomplish Posting. Transferring a message from the Message Transfer System to a recipient's User Agent is called Delivery; the recipient's User Agent and Message Transfer System engage in a Delivery Protocol in order to accomplish Delivery.

The point at which responsibility for a message is transferred is called a Slot. The Posting Slot is the point at which responsibility for a message passes from an originator's User Agent to the Message Transfer System; the Delivery Slot is the point at which responsibility for a message passes from the Message Transfer System to a recipient's User Agent.

The model divides messages into two parts, the message content and the message envelope. The message content is the information that the originator wishes to send to the recipient; this message format specification deals solely with the message content. The message envelope consists of all the information necessary for the Message Transfer System to do its job; this message format specification does not specify the message envelope. Some of the data appearing on the message envelope could be redundant with some data found in the message content. The Message Transfer System is not expected to examine the message content unless it is told to do so by the originator's or recipient's User Agent.

This message format specification places no restrictions on the Message Transfer System itself, except that it be able to transfer messages between originating and receiving UAs without reading or altering the contents of messages unless otherwise instructed by the UAs. In addition, this message format specification does not dictate the form or nature of any protocol used by the Message Transfer System. Finally, this message format specification does not specify the content or form of the message envelope. That is, the message format specification defines the format for the contents of messages, not the manner in which they are transmitted.

Many of today's commercially available CBMSs incorporate all of the facilities represented in the logical model. Their architectures may reflect the economies that can be taken when implementing systems that are self-contained. For example, stand-alone systems that store messages in a single central

database require no Message Transfer System; an implementation may integrate software for User Agent and Message Transfer System functions, doing away with Posting or Delivery Protocols.

## 2.2 Relationship to the ISO Reference Model for Open Systems Interconnection

Subcommittee TC97/SC16 of the International Organization for Standardization (ISO) has developed a reference model for describing communications between "open" systems [ISOD-82]. This model is known as the ISO Reference Model for Open Systems Interconnection (OSI). It divides communications protocols into seven layers, ranging from physical interconnection at the lowest layer to data exchange by application programs at the top.

This message format specification deals with data used by an application within a system. Thus, the message format being specified here is not a protocol. Since it is not a protocol, it lies outside of the model for open systems interconnection. User Agents are application layer entities (layer 7), however, and the protocols used by a message transfer system are above the session layer (layer 5).

## 2.3 Messages and Fields

A message is a unit of communication from an originator to a recipient. A message consists of a series of components called fields. Fields can be described according to their meaning in a message (semantics) and according to the format required for them in a message (syntax).

Semantically, a field is just a component of a message; the meanings of particular fields are defined by this message format specification. Syntactically, a field is a unit of data whose form is defined by this message format specification. Additional fields can be defined by users or vendors as long as they conform to the syntactic and semantic rules that this message format specification defines for additional fields.

(A note on terminology: A message consists of components called fields. The words "message" and "field" are used both in the informal sense of the previous sentence and in a more restricted sense as names of particular syntactic elements. As syntactic element names, Message and Field are always capitalized.)

Some CBMS functions are based on the contents of particular fields; other functions (such as the ability to read a message) may have little to do with the fields themselves. Section 3.2 discusses some of the specific functions that a CBMS might provide to users and the fields that must be used to support those functions.

## 2.4 Message Originators and Recipients

This message format specification refers to message originators and recipients. These terms were defined functionally in Figure 1. When the message format specification refers to the identity of a message originator or recipient, it means "that information which uniquely identifies the message originator or recipient within the domain of the given message system." The syntax and semantics of message addressing are not within the scope of the message format specification.

Originators and recipients can be people, roles, processes or groups.

People. People as originators and recipients are specific individuals.

Roles. Roles identify functions within organizations as opposed to the specific individuals who perform them. For example, consider a newspaper that produces both morning and evening editions and therefore operates with more than one shift. Someone wishing to contact the city desk would send a message to the city desk role rather than trying to determine exactly who was assigned to the city desk at a specific time. (Of course, messages can usually be sent to the individuals directly whether or not they are actually performing a role at the time.)

Processes. A process in a computer could serve as either an originator or a recipient for messages. A computer system might originate a message to notify a recipient about the status of some task. For example, an archive utility could notify users about files that have been archived; a distributed file system could notify a user that a remote file has been deposited on a local file system. Messages could be used by computer systems to warn about some impending condition or even to monitor the performance of the computer itself. Some computer processes may also be message recipients, taking action based upon message contents.

In addition, some CBMSs allow messages to be sent to groups. A group is a predefined list of message recipients. Using a

group name as a recipient permits message originators to designate a potentially large number of recipients using a single recipient identifier. This makes using the CBMS more convenient and accurate.

# 3. SEMANTICS

This section discusses two major topics, message processing functions and message field meanings. Section 3.1 describes the six functional groups of message fields. The functional groups are Origination, Dates, Recipients, Cross-referencing, Message-handling, and Message-contents. They are explained more fully in Section 3.1.1, along with detailed discussion of the semantics of all the fields in each functional group. Section 3.2 describes message processing functions whose operation is based on the meanings of particular message fields.

## 3.1 Semantics of Message Fields

The definition of a message is discussed generally in Sections 1 and 2. Semantically valid messages must contain one From field, one To field, and one Posted-Date field. They may contain, in addition, any number of other fields, depending on the processing and functions supplied by the originating or receiving CBMS. (Section 3.2 describes classes of functions supplied by CBMSs.)

### 3.1.1 Types of fields

Message receiving programs are required to interpret fields according to the semantics described in the remainder of this section. The message fields defined in this document are grouped into the following functional categories.

- Originator fields indicate who or what participated in the creation of the message and where replies should be directed. (See Section 3.1.3.)
- Date fields record when events take place, for a variety of events, such as message creation or expiration. (See Section 3.1.5.)
- Recipient fields indicate who or what is intended to receive a message. (See Section 3.1.4.)
- Cross-reference fields label a message or refer to other messages. (See Section 3.1.6.)
- Message-handling fields record the type of service a

message's sender requested of a message transfer system or indicate how the message should be treated by its recipients. (See Section 3.1.7.)

- Message-content fields either contain the primary content of a message, or index the message, or summarize the message. (See Section 3.1.8.)

- Extension fields provide mechanisms for extending the message format specification. (See Section 3.1.9.)

### 3.1.2 Semantic Compliance Categories

For purposes of determining whether a CBMS complies with the semantic requirements of this message format specification, message fields have been divided into three categories:

REQUIRED These fields must be present in all messages and must be processed by message receiving programs as defined by the message format specification.

BASIC These fields need not be present in all messages but when they do appear, they must be processed by message receiving programs as defined by the message format specification.

OPTIONAL These fields need not be present in all messages and may be ignored by message receiving programs. The exact meaning of "ignored" is not specified by the message format specification. However, a CBMS must recognize the existence of an optional field (that is, optional fields should not cause errors) and must not process the field in a manner contrary to the semantics defined for that field by the message format specification. It is left to the discretion of a recipient's CBMS what action is to be taken when an instance of a locally unimplemented optional field is detected.

(Syntactic compliance is defined in Section 4.1.2.)

### 3.1.3 Originator fields

A message originator may be a person, role, or process. Originator fields identify a message's author, who is responsible for the message, who or what sent it, and where any replies should be directed. (See Section 2.4.)

From (REQUIRED)

This field contains the identity of the originator(s) taking formal responsibility for this message. The contents of the From field is to be used for replies when no Reply-to field appears in a message.

Reply-To (BASIC)

This field identifies any recipients of replies to the message.

Author (OPTIONAL)

This field identifies the individual(s) who wrote the primary contents of the message. Use of the Author field is discouraged when the contents of the Author field and the From field would be completely redundant.

Sender (OPTIONAL)

This field identifies the agent who sent the message. It is used either when the sender is not the originator responsible for the message or to indicate who among a group of originators responsible for the message actually sent it. Use of the Sender field is discouraged when the contents of the Sender field and From field would be completely redundant. The sender field may specify only one originator identity and appear only once in a message.

### 3.1.4 Recipient fields

Message recipients may be people, roles, processes, or groups. (See Section 2.4). Recipient fields identify who or what is to receive the message.

To (REQUIRED)

This field identifies the primary recipients of a message.

Bcc (OPTIONAL)

This field identifies additional recipients of a message (a "blind carbon copies" list). The contents of this field are not to be included in copies of the message sent to the primary and secondary recipients. See section 3.2.1 for further discussion of the use of blind carbon copies lists.

Cc (BASIC)

This field identifies secondary recipients of a message (a "carbon copies" list).

Circulate-Next (OPTIONAL)

This field is used in conjunction with the Circulate-To field. (See Section 3.2.6.1.) It identifies all recipients in a circulation list who have not received the message.

Circulate-To (OPTIONAL)

This field identifies recipients of a circulated message. (See Section 3.2.6.1.) It is used in conjunction with the Circulate-Next field.

### 3.1.5 Date fields

Date fields for two kinds of uses are provided. Dates can be associated with some event in the history of a message and dates can delimit the span of time during which the message is meaningful (its life span).

Posted-Date (REQUIRED)

This field contains the posting date, which is the point in time when the message passes through the posting slot into a message transfer system. Only one Posted-Date field is permitted in a message.

Date (OPTIONAL)

This field contains a date that the message's originator wishes to associate with a message. The Date field is to the Posted-Date field as the date on a letter is to the postmark added by the post office.

End-Date (OPTIONAL)

This field contains the date on which a message loses effect. (See also Section 3.2.5.)

Received-Date (OPTIONAL)

This field is also called Delivery date. This field may be added to a message by the recipient's message receiving program. It indicates when the message left

the delivery system and entered the recipient's message processing domain.

Start-Date (OPTIONAL)

This field contains the date on which a message takes effect. (See also Section 3.2.5.)

Warning-Date (OPTIONAL)

This field is used either alone or in conjunction with an End-Date field. It contains one or more dates. These dates could be used by a message processing program as warnings of an impending end-date or other event. (See also Section 3.2.5.)

### 3.1.6 Cross-reference fields

Cross-reference fields can be used to identify a message and to provide cross-references to other messages. (See Section 3.2.4.)

In-Reply-To (OPTIONAL)

This field designates previous correspondence to which this message is a reply. The usual contents of this field would be the contents of the Message-ID field of the message(s) being replied to.

Message-ID (OPTIONAL)

This field contains a unique identifier for a message. This identifier is intended for machine generation and processing. Further definition appears in Section 3.2.4.1. Only one Message-ID field is permitted in a message.

Obsoletes (OPTIONAL)

This field identifies one or more messages that this one replaces.

Originator-Serial-Number (OPTIONAL)

This field contains one or more serial numbers assigned by the message's originator. Messages with multiple recipients should have the same value in the Originator-Serial-Number field.

References (OPTIONAL)

This field identifies other correspondence that this message references. If the other correspondence contains a Message-ID field, the contents of the References field must be the message identifier.

### 3.1.7 Message-handling fields

Message-handling fields describe aspects of how a message is to be handled or categorized.

Precedence (OPTIONAL)

This field indicates the precedence at which the message was posted. Ordinarily, message precedence or priority is a service request to a message transfer system. A message originator, however, can include precedence information in a message. One example of precedence categories are those used by the U.S. Military: "ROUTINE," "PRIORITY," "IMMEDIATE," "FLASH OVERRIDE," and "EMERGENCY COMMAND PRECEDENCE."

Message-Class (OPTIONAL)

This field indicates the purpose of a message. For example, it might contain values indicating that the message is a memorandum or a data-base entry.[1]

Reissue-Type (OPTIONAL)

This field is used in conjunction with message encapsulating (see Section 3.2.2) to differentiate between messages being assigned or redistributed.

Received-From (OPTIONAL)

This field contains a record of a message's path through a message transfer system. The recipient's message receiving program could store here any information about the transfer that it obtained from a message transfer system.

---

[1] The message format specification is not intended to be used as a specification for exchanging data-base records. Messages, however, sometimes contain data from or for a database.

### 3.1.8 Message-content fields

The intent of most messages is to communicate some particular information from originator to recipient. Several fields in a message are designed to contain that information.

Subject (BASIC)

This field contains any information the originator provided to summarize or indicate the nature of the message.

Text (BASIC)

This field contains the primary content of the message.

Attachments (OPTIONAL)

This field contains additional data accompanying a message. It is similar in intent to enclosures in a conventional mail system.

Comments (OPTIONAL)

This field permits adding comments to the message without disturbing the original contents of the message.

Keywords (OPTIONAL)

This field contains keywords or phrases for use in retrieving a message.

### 3.1.9 Extensions

This message format specification allows two additional types of fields, vendor-defined fields and as-yet-undefined (extension) fields that will be introduced by extensions to this message format specification.

vendor-defined-field

Any field not defined in this message format specification or any extension or successor to it is a vendor-defined field. Names for vendor-defined fields could be preempted by extensions to this message format specification.

extension-field
: Any field that is defined in a document published as a formal extension or replacement to this message format specification.

## 3.2 Message Processing Functions

A CBMS provides three basic classes of functions: creating messages, transmitting messages to their recipient, and post-receipt processing. Although the message format specification does not define the number or nature of user functions in CBMSs, the meanings for the fields clearly assume certain kinds of functions. For example, fields specifying recipients of replies to messages assume some kind of reply function; fields specifying message life span assume some kind of date processing functions.

This section provides more detail on the processing that might be done by these kinds of functions, discussing the message fields that would be used and how they would be used. (See summary in Table 1.)

| Processing Function | Fields Involved |
|---|---|
| Message creation and posting | Author, From, Sender, To, Cc, Bcc |
| Message reissuing | Reissue-Type |
| Reply generation | Reply-To |
| Cross-referencing | Message-ID, In-Reply-To, References, Obsoletes, Originator-Serial-Number |
| Life span functions | Start-Date, End-Date, Warning-Date |
| Recipient processing | Circulate-To, Circulate-Next |

TABLE 1. FIELDS USED IN MESSAGE PROCESSING FUNCTIONS

### 3.2.1 Message creation and posting

Messages can be created either by reissuing an existing message to a new recipient (see Section 3.2.2) or by creating a new message. The process of message creation might mean that some fields of a new message are filled in from the contents of some other message. Reply functions (Section 3.2.3) provide an example of this.

Different individuals could be involved in different phases of originating a message: creating it, taking responsibility for it, and explicitly interacting with a CBMS to send it to its recipient. One or more individuals may create a message (that is, write, but not necessarily enter it into the CBMS); they are said to be the message's authors, identified by the Author field. One or more individuals may take responsibility for its contents and the decision to post it; they are identified by the From field. One individual explicitly posts a given message; this person is called the message's sender (identified by the Sender field).

The sender and author(s) are often, but not always, responsible for the message. A common case in which the sender is not responsible for the message is when a secretary enters and posts messages for someone else. An example of a situation in which a message's author is not responsible for the message itself is when an administrative assistant prepares a report that is sent under a manager's signature.

The use of the Cc field is identical to current business practice. This field contains the formal secondary recipients of the message.

Messages containing Bcc fields are treated specially by CBMSs. The contents of this field are not included in copies of the message sent to the recipients other than the originator who are not included in the Bcc field itself. Some systems include the contents of the Bcc field only in the originator's copy; others include all or part of the Bcc field in the copies sent to the recipients indicated in the Bcc field. This specification does not indicate exactly how the Bcc field is to be treated.

### 3.2.2 Message reissuing and forwarding

Reissuing and forwarding both serve the general user goal of passing a message on to a new set of recipients. Forwarding is the term used for an informal mechanism, which CBMSs implement by copying some or all of the original message into the contents of a field in the new message. Reissuing is the term used for a formal mechanism to ensure that the message being passed on never loses its integrity as a previously sent message. CBMSs use reissuing to implement several different functions, depending on the purposes being served:

- Redistribution. Making others aware of the complete and unaltered contents of the message.

- o Assignment. Delegating the responsibility for a message to somebody else.

These purposes are exemplified in Figure 2.

When a CBMS examines a forwarded message, it cannot always distinguish the old message from what was added when the forwarding took place. In addition, the forwarded information might no longer have the form of a message. This is usually because the format of the message has been changed (for example, to pure unformatted text). (See Figure 2 for an example of how a CBMS might forward a message.) In contrast, a reissued message can always be separated from its enclosing message and never loses its identity as a correctly formed message.

This specification provides the Reissue-Type field for supporting reissuing. Forwarding, since it is an informal means of serving the purpose of passing on information, has no supporting fields in the specification.

This specification provides for reissuing of messages by encapsulating. This method embeds the entire original message inside a new message. Encapsulating adds structure around the message [2]. This allows any part of it to be easily extracted.

This procedure for passing on previously sent messages is a matter of organizational policy and has authentication as an associated issue. Each organization must decide if the CBMS it acquires should support reissuing or simply supply forwarding.

### 3.2.2.1 Redistribution

Redistribution is a CBMS function for sending the original contents of a message intact and unchanged to new recipients. A redistributed message is identical to the original message with the exception of added information about the reissuing. For reissuing with this purpose, the Reissue-Type field contains the ASCII string "Redistribution." The original message has been included directly in a new message. (See Figure 2.)

---

[2] A message can contain another message, and that message can contain another message, and so on to any depth of encapsulating. This can occur by reissuing a message repeatedly.

The Original Message

John Doe wishes Jane Jones to get a copy of the following message:

```
Message:
  Field: From "Jean Smith"
  Field: Posted-Date "27 January 1983"
  Field: To "John Doe"
  Field: Subject "Next Project Meeting"
  Field: Text "The agenda for ..."
```

Redistribution

```
Message:
  Field: From "John Doe"                        John Doe is responsible
  Field: Posted-Date "28 January 1983"          for the redistribution.
  Field: To "Jane Jones"
  Field: Reissue-Type "Redistribution"          This message directly
  Message:                                      incorporates a
    Field: From "Jean Smith"                    redistributed message.
    Field: Posted-Date "27 January 1983"
    Field: To "John Doe"
    Field: Subject "Next Project Meeting"
    Field: Text "The agenda for ..."
```

Forwarding

```
Message:
  Field: From "John Doe"
  Field: Posted-Date "28 January 1983"
  Field: To "Jane Jones"
  Field: Text                                   A realization of the
    "From Jean Smith                            original message is
     To John Doe                                copied into the Text field.
     Sent on 27 January 1983                    Note that John's CBMS
     Subject Next Project Meeting               has chosen to represent
                                                it as a text string.
     The agenda for ..."
```

FIG. 2. MESSAGE FORWARDING AND REDISTRIBUTION

### 3.2.2.2 Assignment

Assignment is the process of designating responsibility. In some organizations, formal message traffic is distributed to one or more parts of the organization (called offices) where it is directed to the appropriate individuals or other offices for final disposition. Assignment is done by reissuing a message with the Reissue-Type field containing the ASCII string "Assigned." A message which contains this field is to be interpreted as meaning that the addressees in the "To" field have had the reissued message assigned to them for some action. Any addressee in the "Cc" field has had the message assigned for information. The "From" field records who assigned the message and the "Posted-Date" field records when the message was assigned.

## 3.2.3 Reply generation

Reply generation involves creating a new message in direct reply to some other message by drawing on the contents of fields in the other message to fill fields in the new message. Many CBMSs provide reply facilities that determine the intended recipients of a reply.

A Reply-To field is defined by this message format specification. When a message contains a Reply-To field, the CBMS should send replies to the recipients designated in the Reply-To field instead of to the recipients designated in the From field. This statement applies to original messages only, not to reissued messages. The message format specification makes no recommendations concerning replies to reissued messages.

Reply-To has several possible applications:

- The individual(s) responsible for the message might not have regular access to a CBMS and would indicate an alternate recipient, for example, a secretary.

- The people responsible for receiving responses might not be the people who were responsible for creating the message.

- Discussion and conference groups could use this feature to ensure correct distribution of any submission by having the conference group itself designated in the Reply-To field.

When the message does not contain a Reply-To field, the recipient should reply to the originators enumerated in the From field. The sender and authors should not be added automatically to the list of those receiving the reply.

Replies could also be sent to the other recipients of the original message. Vendors might offer additional reply facilities, depending on their view of users' organizational requirements.

### 3.2.4 Cross-referencing

A CBMS message may include designator(s) which identify other message(s). The designators are used to refer to related messages so that all information in a chain of correspondence can be determined by a CBMS user. The designator used to identify and cross-reference messages can take either of two forms, unique identifiers or serial numbers.

#### 3.2.4.1 Unique identifiers

Unique identifiers are machine-generated and are intended primarily for processing by computers. While they could be examined by a human user, unique identifiers are not necessarily useful or convenient for people.

Unique identifiers occur in several contexts. They are often used to identify the contents of idual messages unambiguously. When unique identifiers are used this way, they are called message identifiers. Different versions of a message receive new message identifiers; an example of this is reissuing a message with comments.

When a CBMS generates a message identifier, it must be able to guarantee that it is unique, both within the domain of the individual CBMS and globally, across all connected CBMSs. CBMSs could generate globally unique identifiers in several ways, all of which require prior agreement on behalf of the connected CBMSs. One method is to assign each connected CBMS a unique code. A CBMS then generates unique identifiers by using its code as a prefix to some other value that it can guarantee to be unique within its domain. (This second value could be a counter or a timestamp/user-id combination.)

A CBMS can provide functions for tracing chains of correspondence by using unique identifiers. The message format specification defines fields for which a CBMS provides unique identifiers as values. They are Message-ID, References, Obsoletes, and In-Reply-To. (See Section 3.1.6.)

### 3.2.4.2 Serial numbering

Serial numbers are for users to maintain a personal numbering system for messages. The numbers are composed of both letters and digits so that users could maintain several sets of sequences concurrently (for example, A1, A2, A3... and B1, B2, B3...).

Serial numbers are assigned at a defined point in the history of a message. Serial numbers are not unique identifiers; they differ from unique identifiers in that they are not necessarily generated or processed by a CBMS. They are designed to be entered and read by CBMS users. They can be as simple or complex as the user requires. Serial numbers are intended to be used to designate messages about a specific topic, or messages a given user has sent. Serial numbers are intended to be a permanent part of the message, just as unique identifiers are.

A CBMS can provide functions allowing originators to add serial numbers to messages. Originator-Serial-Number is the field provided for an originator to add a serial number to a message before sending it.

## 3.2.5 Life span functions

Messages have life spans, usually delimited by the creation date and the time when the last copy of the message is destroyed. Messages could be meaningless before a certain time or irrelevant after a certain time. For example, a reminder to attend a meeting on 5 June loses most of its value on the sixth; a reminder to attend that same meeting may be of little use on 5 May (although not for the same reason).

A CBMS can define a message's life span explicitly using the Start-Date and End-Date fields. A third field, Warning-Date, when used in conjunction with the End-Date, may be used to signal the approach of the End-Date. Warning-Date may also stand alone and be used by a periodic warning (alarm clock) mechanism.

A CBMS could use these fields to help users manage their message stores. For example, a message whose start date has not yet passed could be bypassed by a retrieval command unless the user requested such messages explicitly. A CBMS could use the end date to help with message store housekeeping either by archiving or deleting the expired messages automatically or by asking the user for some action to be taken on them. The warning date could be used to remind the user automatically of an impending end date, such as a meeting reminder.

### 3.2.6 Requests for recipient processing

Recipients have a wide variety of needs for examining and processing a message, ranging from automatic output on some specified device to the execution of a program embedded in the message itself. Because many of these needs are highly specialized, and support for them not widely implemented, this message format specification does not constrain the requests for processing that may be included in a message.

The message format specification does provide two fields that permit an originator to request circulation list processing from the recipient. These fields are Circulate-To and Circulate-Next.

#### 3.2.6.1 Message circulation

Message circulation involves serial distribution of a message to its recipients, based on a distribution list that is part of the message. The message is delivered first to the first recipient on the distribution list. This recipient, or someone the recipient delegates, sends the message on to the second recipient on the list, perhaps after commenting on or adding to the message. This continues until all recipients on the distribution list have received the message.

This message format specification provides two fields to support message circulation. The Circulate-To field contains the complete distribution list, indicating the full set of recipients, and the Circulate-Next field indicates which recipients have not seen the message. See Figure 3 for an example of message circulation using these two fields.

## 3.3 Multiple Occurrences and Ordering of Fields

Most message fields may occur more than once in a message; the exceptions are the Posted-Date, Sender, and Message-ID fields, which may occur once, at most. What this means is that a received message may contain any number of instances of a particular field (such as the "To" field). If a message contains more than one instance of a particular field, that field "occurs multiply" and that message has "multiple occurrences" of that field.

A particular instance of a message field is not superseded by later instances of the same field. The To field is an example of this.

---

A message originator wishes to circulate a message to recipients A, B and C. The originator includes the following fields in the message:

```
To:              A
Circulate-To:    A, B, C
Circulate-Next:  B, C
```

When recipient A or someone A delegates causes the message to be further circulated, the message is sent to the first address in the Circulate-Next field, and that name is removed from that field:

```
To:              B
Circulate-To:    A, B, C
Circulate-Next:  C
```

B now sends the message on to its final recipient:

```
To:              C
Circulate-To:    A, B, C
```

FIG. 3. EXAMPLE OF MESSAGE CIRCULATION

---

Multiple occurrences of a field are not necessarily equivalent to a single field containing the concatenated contents of the several instances of the given field. For example, with the Text field, concatenating the contents of several instances might lose important distinctions between the contents. A single message could be used to send three different documents, each one in a different Text field. However, putting the three documents into a single Text field would make it much more difficult to extract any individual document.

Encapsulated messages are exceptions to the multiple occurrences rule. For example, the To field in an encapsulated message is not a multiple occurrence of the To field in the enclosing message.

The fields found in a single message may occur in any order. The order in which they occur does not necessarily reflect the

order in which they were created. Nor does it constrain the order in which the message recipient examines, processes, or displays them.

# 4. SYNTAX

This section begins with an introduction to the concepts and elements that constitute the syntax for messages. The second section presents an overview of the encoding scheme. The third section describes in detail the elements of the message syntax.

## 4.1 Introduction

This specification defines syntactic requirements for messages when they are passed from one CBMS to another. The specification is designed to meet the following goals.

- Provide a concise, flexible representation scheme.
- Simplify message parsing.
- Support non-textual components in messages (for example, facsimile, graphics, or speech [3]).

### 4.1.1 Message structure

Messages have two classes of components, fields and messages. A field corresponds to one of the semantic components defined in this message format specification. A message is simply another message.

The type of a field in a message determines both its meaning and the form for its contents. (See Section 4.3.2.)

Fields in a message are composed of syntactic elements called data elements. A Message data element is used to represent messages; a Field data element is used to represent fields. (The term "field" is simply a semantic construct, distinct from "Field Data Element," which is a syntactic

---

[3] While this message format specification is not intended to be used as a basis for the interchange of all facsimile information, it does recognize that CBMS messages may contain facsimile components.

construct.) Many of the fields defined in this message format specification are restricted to containing only one kind of data element. (See Section 4.3.2.)

Each field defined in this message format specification has been assigned a unique numeric identifier that is used in conjunction with the Field data element. Separate identifiers are provided for vendor-defined fields and for extending the identifier encoding space. A list of fields and identifiers appears in Section 4.3.2 and in Appendix C.

Throughout the message format specification, fields are referred to by label name rather than by their numeric identifiers. Field labels are names like "Sender," "Warning-Date," or "Circulate-To." The field labels chosen for the specification are names that are in common use in current CBMSs. The specification does not require a CBMS to use these field labels in displaying fields to the user.

### 4.1.2 Data elements

For the purpose of determining compliance with the syntax defined in this specification, data elements are divided into two groups:

BASIC — All message receiving systems must process these syntactic elements, interpreting their values according to the message format specification.

OPTIONAL — Message receiving systems need not process these syntactic elements in order to be in compliance.

In addition, complying CBMSs must meet requirements regarding their ability to process the components found inside data elements. These requirements are discussed in Section 4.2.2.

This message format specification classifies data element types as either primitives or constructors. Primitive data elements, such as ASCII-String, are basic building blocks. Constructor data elements, such as Message or Sequence, contain one or more primitive or constructor data elements. Some constructors, such as Sequence, may be composed of any other data element. Some, such as Message, may contain only certain data elements. Two data elements, Extension and Vendor-Defined, may be classified as either primitives or constructors, depending on how they are used to extend this specification. The general syntactic form for data elements is discussed in section 4.3.1.

4.1.2.1 Primitive data elements

A primitive data element contains a basic item of information; it is not composed of other data elements. In current CBMSs, the most commonly used primitive data element is ASCII-String,[4] a series of ASCII characters. Other primitive data elements are Integer, 2's complement integers; Bit-String, a series of bits; and Boolean, either True or False.

One primitive data element, End-Of-Constructor, is used only as a structural element within constructor data elements and has no meaning by itself. End-of-Constructor is used to provide an end marker for constructor data elements that do not have an explicit length; any other use is not valid syntactically.

4.1.2.2 Constructor data elements

The Data Element Contents of constructor data elements contain one or more data elements. The most general form of a constructor is a Sequence or a Set, since both Sequences and Sets may contain any data element. Other constructors are specialized forms of sequences.

A Message data element is a constructor. It may contain only Field data elements, other Message data elements, or encrypted or data compressed forms of these elements. A Field data element can contain any data element. It also indicates which specific field is being represented. The contents of some fields are restricted to a single type of data element, such as ASCII-String or Date.

4.1.3 Properties

Any data element may have associated with it a Property-List, which contains properties such as a Printing-Name or one or more Comments. Comment A mechanism to support vendor-defined properties has been supplied by this specification, as well as a mechanism to extend the list of property identifiers.

---

[4] An ASCII-String is not limited to ASCII characters however. The ASCII code table can be extended through standardized techniques as described in FIPS Pub 35, Code Extension Techniques in 7 or 8 Bits [NatB-75].

#### 4.1.3.1 Printing-names

Printing-Names are used to provide labels that can be displayed along with their respective data elements. For example, a message originator may use a Printing-Name property to request that the To field of a message be labeled "Distribution:" when it is printed by its recipients.

#### 4.1.3.2 Comments

The Comment property is used to allow comments to be associated with any data element without affecting its actual contents. For example, someone reviewing the text of a message could add the comment "This looks good" to the Text field without either altering the body itself or adding a separate comment field.

### 4.1.4 Data compression and encryption

Two constructor data elements, Compressed and Encrypted, have been provided for use by a CBMS that supports data compression or encryption. They may be used to hold the compressed or encrypted contents of any data element, including Messages and Fields, and may occur wherever their compressed or encrypted contents may appear. A mechanism is included to allow the user to identify the encryption or compression algorithm used (Sections 4.3.4 and 4.3.5).

## 4.2 Overview of Syntax Encoding

This section provides an overview of the notation and terminology used to represent the syntactic elements (data elements) defined in this message format specification.

All data elements consist of a series of components. Each of the components is composed of a series of 8-bit groups called octets. In this document, the bits are numbered starting from the low-order bit. That is, the low-order (or least significant) bit is called "bit 0" and the high-order (or most significant) bit is called "bit 7."

Five different components may appear in a data element.

- Identifier octet (identifying particular type of data element)

- Length Code (specifying number of octets that appear following it in a data element)
- Qualifier (supplying additional identifying information)
- Property-List component (a Property-List data element containing Property data elements)
- Data Element Contents (containing actual data of the data element)

These components always appear in this order. Not all components are present in all data elements, but the components that are present maintain this relative order.

### 4.2.1 Identifier Octets

The identifier octet is a numeric code containing information that identifies a data element. It is always the first component in a data element. The Identifier octet contains a one-bit flag, indicating whether or not the data element contains a Property-List, and a 7-bit unique identifier for the data element. The value of the data element identifier also indicates whether the data element has a Qualifier.

The most significant bit (Bit 7) of the identifier octet is set to 1 if there are properties associated with the data element; it is set to 0 if there are none. This bit is independent of the remaining seven bits in the identifier octet, which are called the identifier, and provide unique identification for data elements. The associated properties are specified in a Property-List component.

The second most significant bit (Bit 6) of the identifier octet (the most significant bit of the identifier itself) signifies whether or not the data element has a Qualifier. If the bit is set to 1, then the data element has a Qualifier; if it is a 0, the data element does not have a Qualifier. The seven bits of the identifier uniquely identify the data element.

Table 2 shows the settings of the high-order bits of the identifier octet and their associated meaning. Figure 4 demonstrates the bit-level structure of the identifier octet. In this figure, bit 7 is indiciated with P to show its special use.

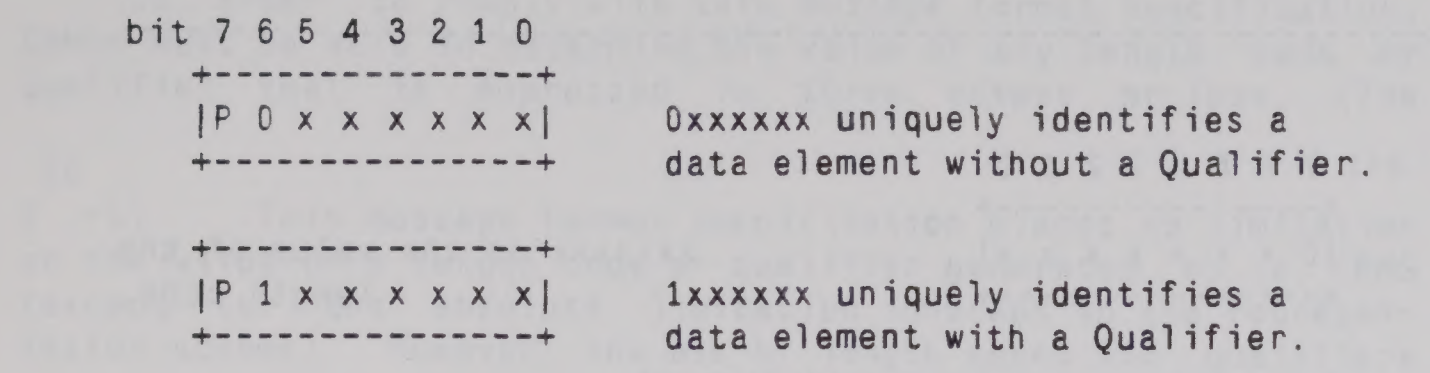

bit 7 6 5 4 3 2 1 0
|P 0 x x x x x x|
0xxxxxx uniquely identifies a data element without a Qualifier.
|P 1 x x x x x x|
1xxxxxx uniquely identifies a data element with a Qualifier.

FIG. 4. STRUCTURE OF IDENTIFIER OCTETS

| Bit | Value | Meaning |
|---|---|---|
| 7 | 0 | The data element does not have properties associated. |
| | 1 | The data element has properties associated. |
| 6 | 0 | The data element does not have a Qualifier. |
| | 1 | The data element has a Qualifier. |

TABLE 2. HIGH-ORDER BITS IN THE IDENTIFIER OCTET

### 4.2.2 Length code and Qualifier components

The Length Code and the Qualifier are both usually one octet in length. They use an encoding scheme that permits extending the component to the size necessary to represent the length of the data element or the value of the Qualifier component.

The most significant bit of the Length Code or Qualifier components determines whether it is one or several octets in length. When the most significant bit is 0, the component is one

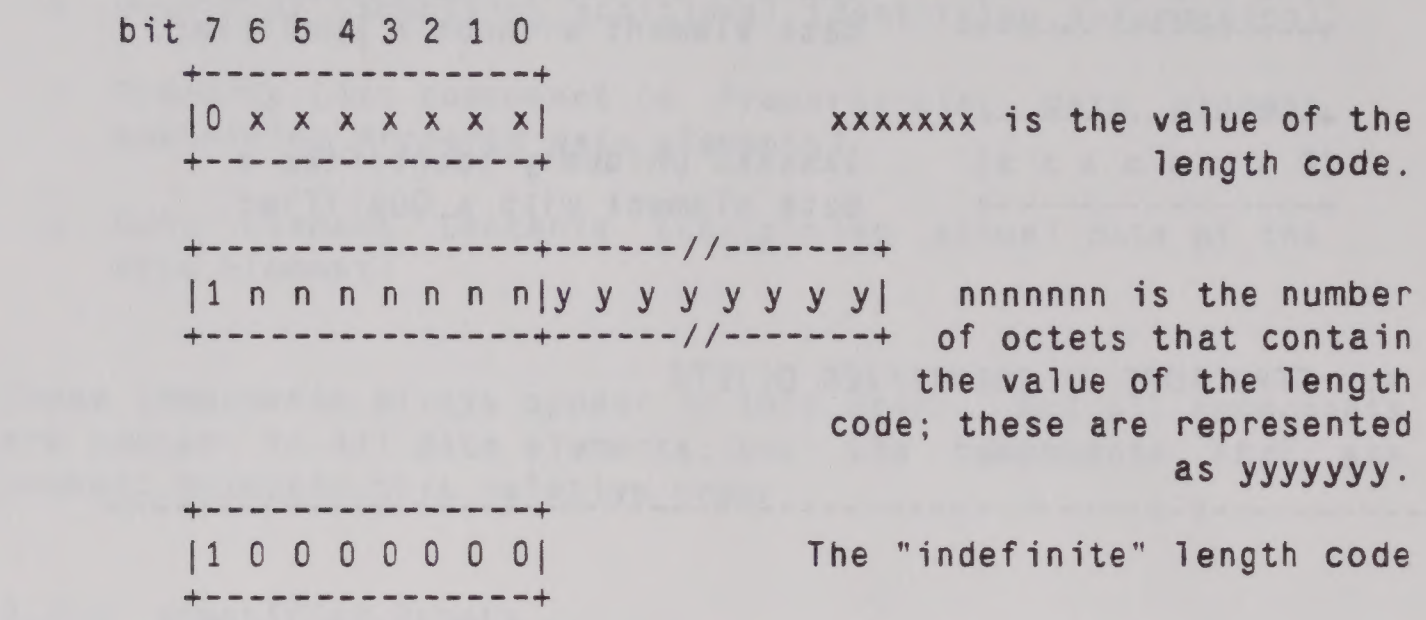

bit 7 6 5 4 3 2 1 0
|0 x x x x x x x|
xxxxxxx is the value of the length code.
|1 n n n n n n n|y y y y y y y y|
nnnnnnn is the number of octets that contain the value of the length code; these are represented as yyyyyyy.
|1 0 0 0 0 0 0 0|
The "indefinite" length code

FIG. 6. REPRESENTATION OF LENGTH CODES

primitive data element [5]. A constructor data element with an indefinite length code has an End-Of-Constructor data element as the last data element in its Data Element Contents. (The length of such a constructor data element is unrestricted, although it must contain at least one data element -- the End-of-Constructor that terminates it -- in its Data Element Contents.)

4.2.2.2 Qualifier

If present, the Qualifier component immediately follows the code component. It is used to provide information essential to the interpretation of the data element contents that is beyond that encoded in the identifier octet or length code. For example, the identifier octet could contain the code for a field, and the Qualifier component would specify what kind of field.

The Qualifier component appears in only a few data elements.

---

[5] This is the result of most primitive elements being able to contain any bit pattern (including the identifier for End-Of-Constructor).

In order to comply with this message format specification, CBMSs must be able to determine the value of any length code or qualifier that is expressed in three octets or less. (The $2^{16}-1$). This message format specification places no limitation on the value of a length code or qualifier generated by a CBMS (except for the absolute limitation inherent in the representation scheme). However, the use of length codes and qualifiers with larger values (particularly values in excess of $2^{32}-1$) should be avoided unless it is known that the receiving system can handle them.

Both Length Codes and Qualifiers have a special convention for dealing with special situations. Length Codes can specify that a data element has indeterminate length; a Qualifier can specify that a data element is implementation defined. These cases are explained further in the next two sections.

### 4.2.2.1 Length Codes

The length code component immediately follows the identifier octet. It is present in every data element. The Length Code indicates the number of octets following it in a data element (that is, excluding the identifier octet and the length code itself). Length Codes appear in one of three formats: short, long, and indefinite.

A short Length Code is one octet long. Its most significant bit (Bit 7) is set to 0 and its value is in the range 0 through 127.

A long Length Code is at least two octets long. The first octet always has its most significant bit (Bit 7) set to 1. The other seven bits of this octet contain the number of octets making up the rest of the Length Code, and these octets contain $(2^{1016}-1)$ (that is, 127 octets to represent the value).

An indefinite Length Code is one octet long. Its most significant bit (Bit 7) is set to 1 and its other bits are all 0. (See Figure 6.) An indefinite Length Code may appear only as part of a constructor data element; it may not occur in a

octet in length. When the most significant bit is 1, the other seven bits of the first octet encode the number of octets in the rest of the component. The actual value begins in the next octet and is interpreted as an unsigned integer.

A single octet is sufficient for most Length Code and Qualifier components. For those cases where the value of the Length Code or the Qualifier must be greater than 127, extra octets can be added, up to a maximum of 127 octets. Figure 5 shows the encoding scheme, as well as an example of a value less than 127 and one greater than 127.

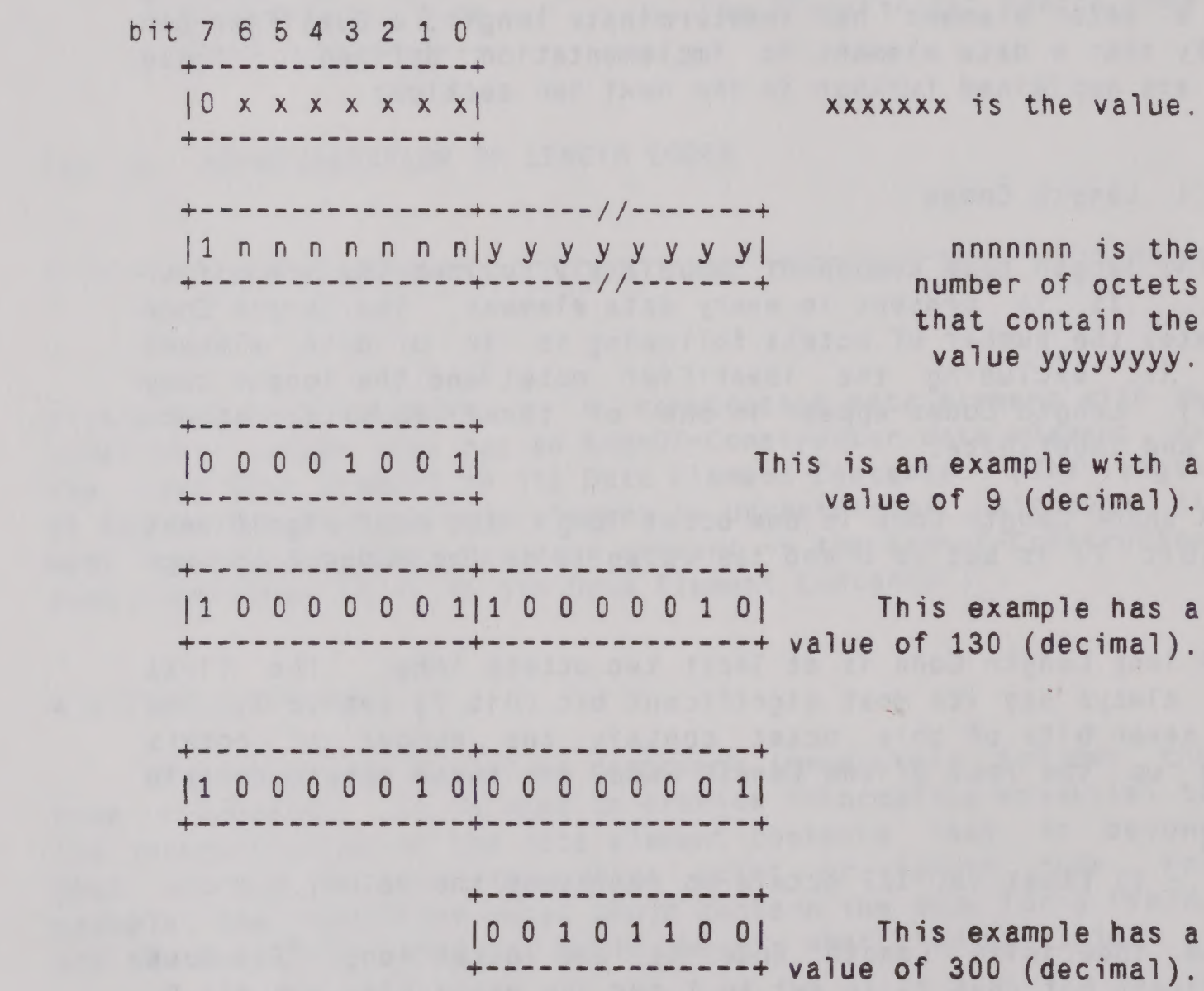

bit 7 6 5 4 3 2 1 0
|0 x x x x x x x|
xxxxxxx is the value.
|1 n n n n n n n|y y y y y y y y|
nnnnnnn is the number of octets that contain the value yyyyyyyy.
|0 0 0 0 1 0 0 1|
This is an example with a value of 9 (decimal).
|1 0 0 0 0 0 0 1|1 0 0 0 0 0 1 0|
This example has a value of 130 (decimal).
|1 0 0 0 0 0 1 0|0 0 0 0 0 0 0 1|
|0 0 1 0 1 1 0 0|
This example has a value of 300 (decimal).

FIG. 5. ENCODING MECHANISM FOR QUALIFIERS AND LENGTH CODES

In the Bit-String data element, it indicates the number of unused bits in the final octet of the Data Element Contents. In the Field and Property data elements, it indicates which field or property the data element represents. In the Compressed and Encrypted data elements, it indicates which compression or encryption algorithm has been used. In the Message data element, it indicates the type of message.

The length of the Qualifier component depends on the encoding of the Qualifier. (See Figure 7.) A short Qualifier is one octet long. Its most significant bit is 0 and its value is in the range 0 through 127. A long Qualifier is at least two octets in length. The most significant bit is always 1 and the other 7 bits indicate the number of octets in the value of the Qualifier.

---

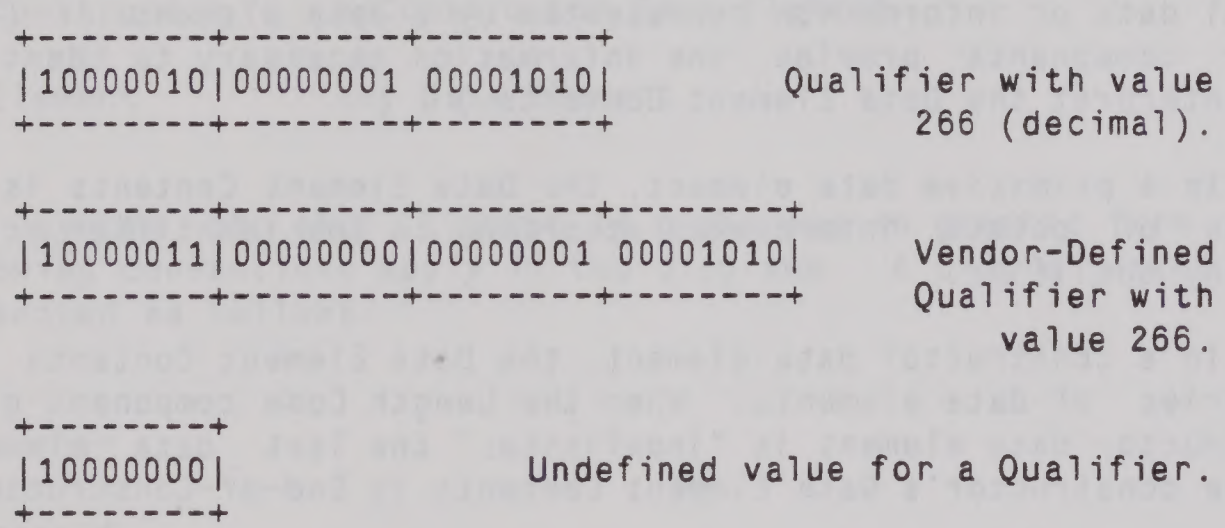



FIG. 7. EXAMPLES OF QUALIFIER VALUES

---

This message format specification allows implementations to define their own values for Qualifiers. A vendor-defined Qualifier is any long Qualifier in which the first octet in the value is 0. The value used to identify this Qualifier is not guaranteed to be unique and the same value may be used by different implementations to define different Qualifiers.

### 4.2.3 Property-List

A Property is an attribute being associated with, but not essential to the interpretation of, a data element. The properties currently defined by this message format specification are Printing-Name and Comment. A Property-List component of a data element is represented by a Property-List data element that in turn contains Property data elements.

A data element contains at most one Property-List. The most significant bit in the identifier octet of the data element indicates whether a Property-List is present.

### 4.2.4 Data Element Contents

The Data Element Contents component of a data element is the actual data or information represented by a data element. (The other components provide the information necessary to identify and interpret the Data Element Contents.)

In a primitive data element, the Data Element Contents is a series of octets interpreted according to the identifier octet and any qualifier.

In a constructor data element, the Data Element Contents is a series of data elements. When the Length Code component of a constructor data element is "indefinite," the last data element in the constructor's Data Element Contents is End-of-Constructor.

The length of the Data Element Contents (in octets) is the difference between the value of the Length Code and the sum of the following:

- the length of the Qualifier component (depends on the data element)
- the length of the Property-List component.

## 4.3 Data Element Syntax

This message format specification defines nineteen (19) different data elements. Section 4.3.1 defines the encoding form for data elements in general and the syntax for each data element. Section 4.3.2 describes the use of specific data

elements as part of the Data Element Contents of a Field data element. A summary of the syntactic form appears in Appendix F; summaries of the data element syntax appear in Appendix G.

### 4.3.1 Data elements

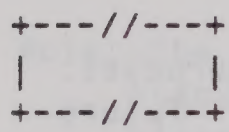
This section presents the general syntactic form for all data elements defined by this message format specification and the detailed syntax for each data element. The data elements are presented by syntactic class: primitive data elements (Section 4.3.1.1), constructors (Section 4.3.1.2), and data elements which can be either (Section 4.3.1.3).

For convenience, the following terminology is used in this section.

| Term | Meaning |
|---|---|
| Primitive | a Primitive Data Element |
| Constructor | a Constructor Data Element |
| Element | any Data Element |

The syntax of each Element is presented in graphic form. The following conventions apply in the diagrams. A single octet is represented as follows.

Components that vary in length are represented as follows.



Each Element has up to five components: an Identifier, a Length Code, a Qualifier, a Property-List, and the Data Element Contents.

In the diagrams, the contents of the identifier octet is

shown as a "P" followed by an identifier represented in binary. (See Figure 4.)

A length code is always represented in the following manner:

```
+---//---+
|Lxxxxxxx|
+---//---+
```

A qualifier is always represented in the following manner:

```
+---//---+
|Qxxxxxxx|
+---//---+
```

A Property-List (if present) always immediately precedes any occurrence of Data Element Contents.

The Data Element Contents appears in diagrams as one of the following:

- "element(s)", which may be any data element(s)
- "anything," which is undefined and may be any combination of bits
- a specific data element
- the interpretation to be applied to the bits within the octets that constitute the element (such as ASCII or Integer)

Two data elements have been reserved for special purposes. The Extension data element is provided to allow for future expansion of the possible data elements. The Vendor-Defined data element allows CBMS vendors to define their own data elements. Vendor-Defined data elements are not guaranteed to be unique, since two implementations could define different data elements using the same identifier. Vendor-Defined data elements should be used and interpreted by prior agreement.

In the following sections, each element is presented with its name, compliance classification (BASIC or OPTIONAL), its identifier (both in hexadecimal and in octal), a brief description of its use, and a graphic representation. Each data element description has the following form.

---

Data Element Name | (Compliance) ( Category ) | identifier octet 16 | identifier octet 8

Description of the syntax of the data element.

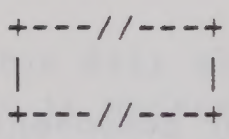


Diagram representing data element

---

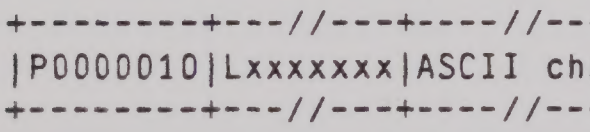
#### 4.3.1.1 Primitives

The data elements in this section are arranged in alphabetical order by name. (Appendix C presents the identifiers in numeric order.)

ASCII-String (BASIC) 02 (16) 002 (8)

This data element contains a series of ASCII characters [NatB-80], each character right-justified in one octet. For 7-bit ASCII characters, the most significant bit of each octet must be 0.

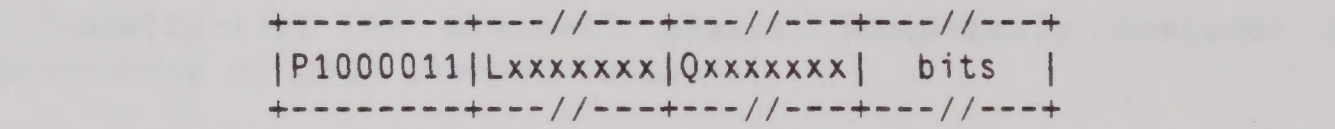
Note: The ASCII code table can be extended through standardized techniques [NatB-75] to introduce additional 7-bit or 8-bit characters or additional code tables.



Bit-String (OPTIONAL) 43 103
16 8

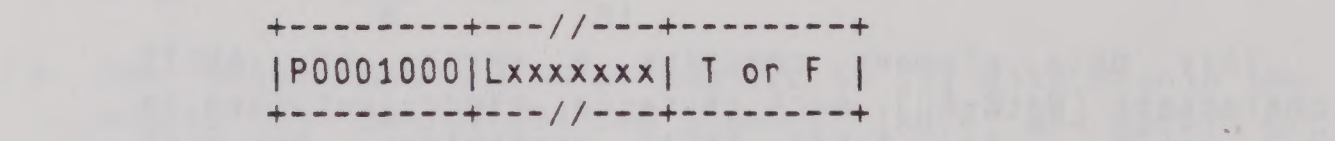
This data element contains a series of bits. It uses the Qualifier data element component to record the number of bits of padding (as an eight bit unsigned integer) needed to fill the final octet of the Data Element Contents to an even octet boundary. These padding bits have no meaning and occur in the low order bits of the final octet. The valid values for the Qualifier component are 0 through 7. The number of bits in the Data Element Contents is calculated from the following formula.

8 * number of octets in the Data Element Contents - value of Qualifier component



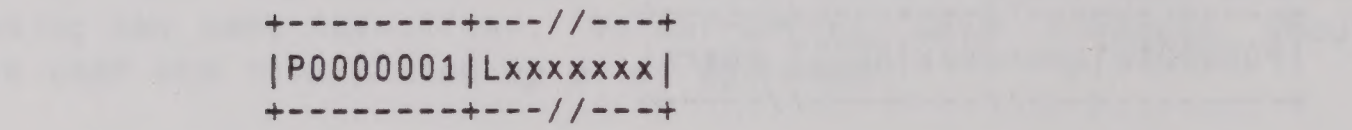

Boolean (OPTIONAL) 08 010
16 8

This data element contains one octet whose value is either true or false. False is represented by all bits being 0; true is represented by all bits being 1 (although any non-zero value should be interpreted as true).



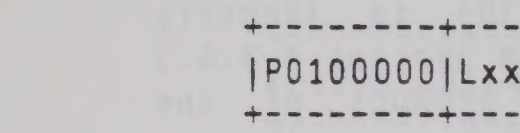

End-of-Constructor (BASIC) 01 001
16 8

This data element terminates the Data Element Contents in a constructor data element that has indefinite length. This data element has no Contents component. (Use of this element is described in Section 4.2.2.1.)



Integer (OPTIONAL) 20 040
16 8

This data element contains a 2's complement integer of variable length, high order octet first. It is recommended that the data element contents be either 2 or 4 octets long whenever possible.

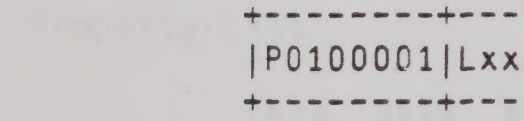



No-Op (OPTIONAL) 00 000
16 8

This data element does nothing. No-Op is used whenever it is necessary to include a data element that means "no operation." It is a short placeholder.

Padding (OPTIONAL) 21 041
16 8

This data element is used to fill any number of octets. The contents of a Padding element are undefined and convey no information.

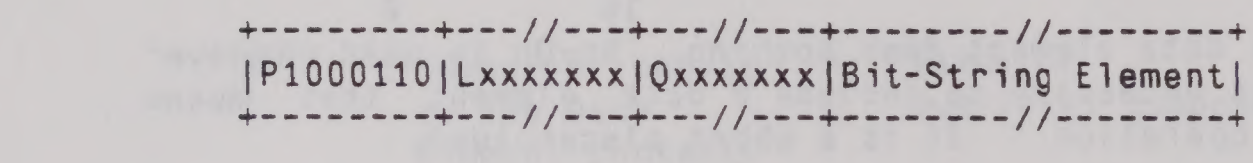



4.3.1.2 Constructors

The data elements in this section are arranged in alphabetical order.

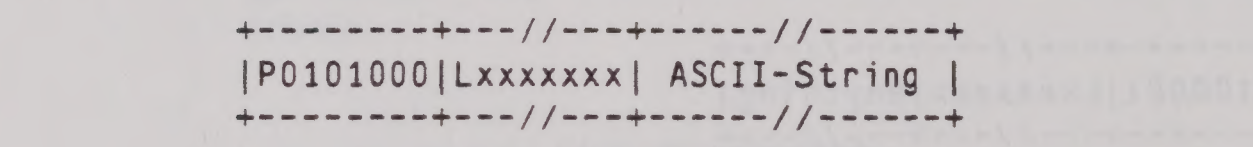
Compressed (OPTIONAL) 46 106
16 8

This data element must contain a Bit-String data element. It is used to represent any data that has been compressed; it may be used wherever its uncompressed contents may appear. A Qualifier data component appears in each Compressed data element; it contains a compression identifier (CID) to identify the compression algorithm used. (See Section 4.3.5.) The Data Element Contents contains the product of the compression process.



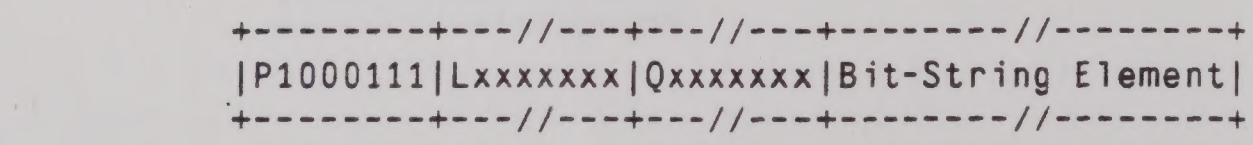

Date (BASIC) 28 050
16 8

This data element contains an ASCII-String data element, which is a representation of a date and time formatted in accordance with PUBS 4 [NatB-68], 58 [NatB-79a] and 59 [NatB-79b]. The use of time and time zone is optional. It is recommended that numeric offsets be used to indicate time zone rather than alphabetic abbreviations.



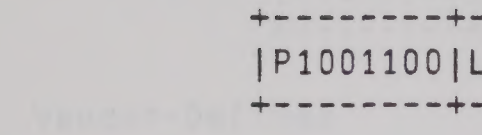

Encrypted (OPTIONAL) 47 107
16 8

This data element must contain a Bit-String. It is used to represent any data that has been encrypted; it may be used wherever its unencrypted contents may appear. A Qualifier data component appears in each Encrypted data element; it contains an encryption identifier (EID) identifying the encryption algorithm used. The Data Element Contents is the product of the encryption process.



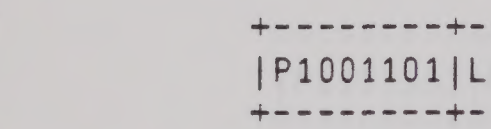
Field (BASIC) 4C 114
16 8

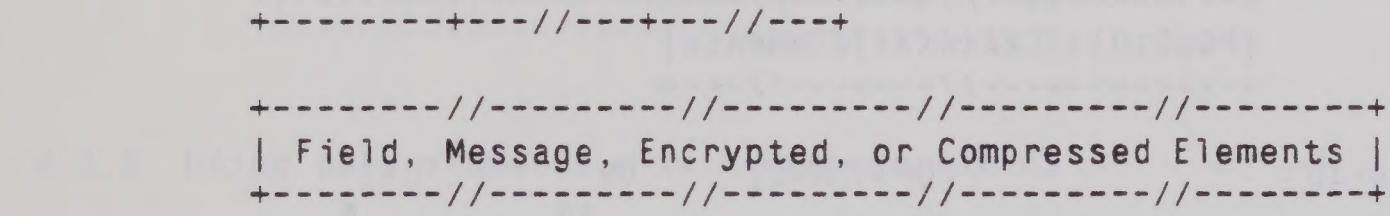
This data element uses a Qualifier data element component. The Qualifier component contains a Field Identifier (FID) indicating which specific field is being represented.




Message (BASIC) 4D 115
16 8

This data element may contain Field or Message data elements. Its Qualifier component contains a Message type (MID) indicating the type of the message. (The MID is completely different from the message identifier in the Message-ID field and should not be confused with it.)



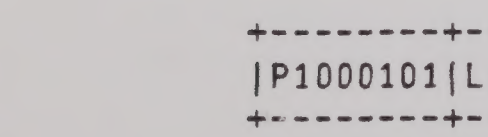




Property-List (OPTIONAL) 24 044
16 8

This data element contains a series of Property data elements to be associated with another data element.

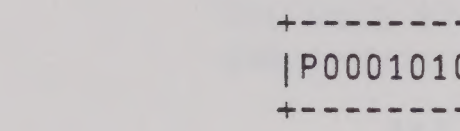
Property (OPTIONAL) 45 105
16 8

This data element uses a Qualifier data element component. The Qualifier component contains a Property-Identifier (PID) to indicate which specific property is being represented.



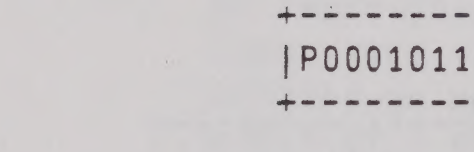
Sequence (OPTIONAL) 0A 012
16 8

This data element contains any series of data elements. Sequence differs from Set in that the data elements making up the Data Element Contents must be considered as an ordered sequence (according to their order of appearance in the sequence.)



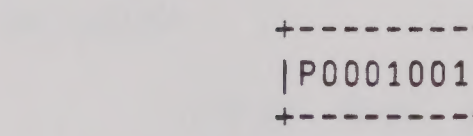

Set (OPTIONAL) 0B 013
16 8

This data element contains any series of data elements with no ordering of the elements implied. (Sequence provides an ordered series.) Although the data elements contained in a Set must be stored sequentially, the order in which they are stored is not defined and not processed.




Unique-ID (OPTIONAL) 09 011
16 8

This data element is a unique identifier. It need not be human-readable. The Data Element Contents may be an ASCII-String, a Bit-String, or an Integer.

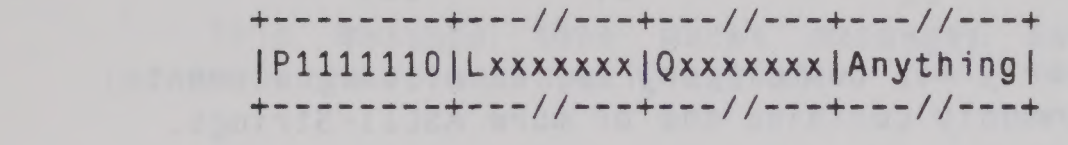



### 4.3.1.3 Data Elements that Extend this Specification

There are two data elements that are used to extend this specification. They can be classified as either primitive or constructor data elements, depending on the extension.

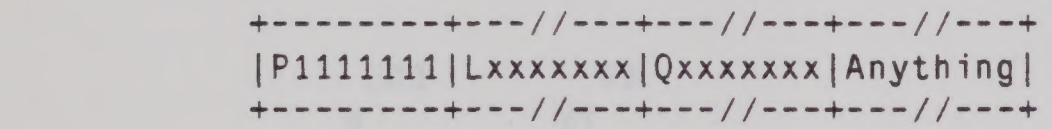
Extension (OPTIONAL) 7E 176
16 8

This data element is used to extend the number of available data elements beyond the 128 that are possible using a 7-bit identifier. A Qualifier component extends the encoding space for identifiers. (Extension and Vendor-Defined have the same syntax.)




Vendor-Defined (OPTIONAL) 7F 177
16 8

This data element is used to represent vendor- and user-defined data elements. A Qualifier component extends the encoding space for identifiers. The Qualifier component is not guaranteed to be unique among all interconnected systems. This data element is interpreted according to prior agreement between systems. (Extension and Vendor-Defined data elements have the same syntax.)




### 4.3.2 Using data elements within message fields

The Data Element Contents of a particular field in a message must contain at least one data element. The types of data elements that can appear in the Data Element Contents of a field are restricted according to what kind of field it is. Appendix A (the master reference appendix for fields) defines which data elements are valid as the Contents for each of the fields.

Some fields have a Data Element Contents that contains "originators" or "recipients." No data element represents the identities of originators or recipients (because that encoding is not within the scope of this message format specification.) These descriptions simply list "originators" or "recipients", implying no restrictions on how the identifiers for originators or recipients are represented.

### 4.3.3 Properties and associated elements

This message format specification defines two properties.

Comment 01 001
16 8

This property may contain any series of data elements; it most commonly contains one or more ASCII-Strings.

Printing-Name 02 002
16 8

This property contains one ASCII-String. In this case, the ASCII-String may contain only the printing ASCII characters plus the "space" character.

### 4.3.4 Encryption identifiers

This message format specification defines two encryption identification codes.

Unspecified 00 000
16 8

Use of this encryption identifier as part of the Encrypted data element indicates that the encryption method being used was not specified for inclusion as part of the data element.

FIPS-Standard 01 001
16 8

Use of this encryption identifier as part of the Encrypted data element indicates that the Federal Information Processing Standard method for data encryption was [NatB-77].

### 4.3.5 Compression identifiers

This message format specification defines one compression identification code for use with the Compressed data element.

Unspecified 00 000
16 8

Use of this compression identifier as part of the Compressed data element indicates that the compression method being used was not specified for inclusion as part of the data element.

### 4.3.6 Message types

This message format specification defines message type (MID) codes for use in classifying the type of a message. The message type could be confused with the message identifier in the Message-Id field; they are completely distinct concepts.

FIPS-Standard 01 01
16 8

This message type marks messages defined by this message format specification.

## SUMMARY OF APPENDIXES

Appendix A Defines the fields in the message format specification. This alphabetical appendix is for reference use by implementors. It contains semantic definitions of fields from Section 3.1. It also defines Field Identifier values and specifies which data elements are valid as the Contents for each of the fields.

Appendix B Defines the data elements in the message format specification. This alphabetically ordered appendix is for reference use by implementors. It consolidates information from Section 4.3.

Appendix C Provides a reference table listing the data elements in numerical order by their identifier octets.

Appendix D Provides a reference table summarizing the components of messages according to whether they are required or optional for CBMSs implementing the specification.

Appendix E Provides a reference table organizing the message components according to the functional class of the components.

Appendix F Provides an overview of the syntactic elements defined by this message format specification.

Appendix G Summarizes syntactic elements according to whether they are required or optional for a CBMS implementing the message format specification.

Appendix H Examples of each syntactic element displaying their syntax and describing their associated semantics.

# APPENDIX A
# FIELDS -- IMPLEMENTORS' MASTER REFERENCE

This appendix defines all of the fields in the message format specification for reference use by implementors. It contains semantics definitions of fields from Section 3.1. It also defines Field Identifier values and which data elements are valid as the Contents for each of the fields. The field definitions appear alphabetically.

Each field in the list has the following form:

---

Field Name — Compliance — identifier value 16 — identifier value 8

Description of the field semantics. Names of data elements that are valid in the Data Element Contents of this kind of field.

---

Attachments — OPTIONAL — 08 — 010

16 — 8

This field contains additional data accompanying a message. It is similar in intent to enclosures in a conventional mail system. Contents of this field are unrestricted.

Author — OPTIONAL — 0C — 014

16 — 8

This field identifies the individual(s) who wrote the primary contents of the message. Use of the Author field is discouraged when the contents of the Author field and the From field would be completely redundant. This field contains one or more originator identities.

Bcc OPTIONAL 0D (16) 015 (8)

This field identifies additional recipients for a message (a "blind carbon copies list"). The contents of this field are not to be included in copies of the message sent to the primary and secondary recipients. See section 3.2.1 for further discussion of the use of blind carbon copies lists. This field contains one or more recipient identities.

Cc BASIC 06 (16) 006 (8)

This field identifies secondary recipients for a message (a "carbon copies" list). This field contains one or more recipient identities.

Circulate-Next OPTIONAL 0E (16) 016 (8)

This field is used in conjunction with the Circulate-To field. (See Section 3.2.6.1 for further discussion.) It identifies all recipients in a circulation list who have not yet received the message. This field contains one or more recipient identities.

Circulate-To OPTIONAL 0F (16) 017 (8)

This field identifies recipients for a circulated message. (See Section 3.2.6.1 for further discussion.) It is used in conjunction with the Circulate-Next field. This field contains one or more recipient identities.

Comments OPTIONAL 10 (16) 020 (8)

This field permits adding comments onto the message without disturbing the original contents of the message. While the Comments field will usually contain one or more ASCII-Strings, there are no restrictions on its contents.

Date OPTIONAL 11 (16) 021 (8)

This field contains a date that the message's originator wishes to associate with a message. The Date field is to the Posted-Date field as the date on a letter is to the postmark added by the post office. This field contains one Date.

End-Date OPTIONAL 12 022
16 8

This field contains the date on which a message loses effect. (See also Section 3.2.5 for further discussion.) This field contains one Date.

From REQUIRED 01 001
16 8

This field contains the identity of the originators taking formal responsibility for this message. The contents of the From field is to be used for replies when no Reply-to field appears in a message. This field contains one or more originator identities.

In-Reply-To OPTIONAL 13 023
16 8

This field designates previous correspondence to which this message is a reply. The usual contents of this field would be the contents of the Message-ID field of the message(s) being replied to. This field contains one or more Unique-IDs or ASCII-Strings.

Keywords OPTIONAL 14 024
16 8

This field contains keywords or phrases for use in retrieving a message. This field contains one or more ASCII-Strings. (Each keyword or phrase is represented by a separate ASCII-String.)

Message-Class OPTIONAL 15 025
16 8

This field indicates the purpose of a message. For example, it might contain values indicating that the message is a memorandum or a data-base entry. This field contains one data element, an ASCII-String.

Message-ID OPTIONAL 16 026
16 8

This field contains a unique identifier for a message. This identifier is intended for machine generation and processing. Further definition appears in Section 3.2.4.1. Only one Message-ID field is permitted in a message. This field contains one data element, a Unique-ID.

Obsoletes OPTIONAL 26 046
16 8

This field identifies one or more messages that this one supplants. This field contains at least one Unique-ID and may contain more than one.

Originator-Serial-Number OPTIONAL 17 027
16 8

This field contains one or more serial numbers assigned by the message's originator. (Messages with multiple recipients should all have the same value in the Originator-Serial-Number field. This field contains one or more ASCII-Strings. (One ASCII-String is used for each serial number.)

Posted-Date REQUIRED 02 002
16 8

This field contains the posting date, which is the point in time when the message passes through the posting slot into a message transfer system. Only one Posted-Date field is permitted in a message. This field contains one Date.

Precedence OPTIONAL 18 030
16 8

Ordinarily, message precedence or priority is a service request to a message transfer system. A message originator, however, can include precedence information in a message. This field indicates the precedence at which the message was posted. One example of a precedence scheme is the US Military categories "ROUTINE", "PRIORITY", "IMMEDIATE", "FLASH OVERRIDE", and "EMERGENCY COMMAND PRECEDENCE". This field contains one ASCII-String.

Received-Date OPTIONAL 19 031
16 8

This field is also called Delivery date. It may be added to a message by the recipient's message receiving program. It indicates when the message left the delivery system and entered the recipient's message processing domain. This field contains one Date.

Received-From OPTIONAL 1A 032
16 8

This field contains a record of a message's path through a message transfer system. The recipient's message receiving program may store any such information that it obtains from a message transfer system in this field. The contents of this field are unrestricted.

References OPTIONAL 20 040
16 8

This field identifies other correspondence that this message references. If the other correspondence contains a Message-ID field, the contents of the References field must be the message identifier. This field contains one or more Unique-IDs or ASCII-Strings.

Reissue-Type OPTIONAL 25 045
16 8

This field is used in conjunction with message encapsulating (see Section 3.2.2) to differentiate between messages being assigned or redistributed. This field contains one data element, usually an ASCII-String.

Reply-To BASIC 03 003
16 8

This field identifies any recipients for replies to the message. This field contains one or more recipient identities.

Sender OPTIONAL 22 042
16 8

This field identifies the agent who sent the message. It is intended either for when the sender is not the originator responsible for the message or to indicate who among a group of originators responsible for the message actually sent it. Use of the Sender field is discouraged when the contents of the Sender field and From field would be completely redundant. Only one Sender field is permitted in a message. This field contains one originator identity.

Start-Date OPTIONAL 23 043
16 8

This field contains the date on which a message takes effect. (See Section 3.2.5 for further discussion.) This field contains one Date.

Subject BASIC 07 007
16 8

This field contains whatever information the originator provided to summarize or indicate the nature of the message. This field contains one or more ASCII-Strings.

Text BASIC 04 004
16 8

This field contains the primary content of the message. Contents of this field are unrestricted.

To REQUIRED 05 005
16 8

This field identifies primary recipients for a message. This field contains one or more recipient identities.

Warning-Date OPTIONAL 24 044
16 8

This field is used either alone or in conjunction with an End-Date field. It contains one or more dates. These dates could be used by a message processing program as warnings of an impending end-date or other event. (See Section 3.2.5 for further discussion.) This field contains one or more Dates.

# APPENDIX B
# DATA ELEMENTS -- IMPLEMENTORS' MASTER REFERENCE

The appendix defines all of the data elements in the message format specification, for reference use by implementors. It contains no new information but rather consolidates the syntactic information from Section 4.3.

Each data element description has the following form.

---

```
Data Element          (Compliance)   identifier   identifier
    Name              ( Category )    octet          octet
                                         16             8

        Constructive class (primitive or constructor)

        Description of the syntax of the data element.

        +---//---+
        |        |    Diagram representing data element
        +---//---+
```

---

ASCII-String (BASIC) 02 002
16 8

primitive

This data element contains a series of ASCII characters [NatB-80], each character right-justified in one octet. For 7-bit ASCII characters, the most significant bit of each octet must be 0.

Note: The ASCII code table can be extended through standardized techniques [NatB-75] to introduce additional 7-bit or 8-bit characters or additional code tables.

```
+--------+---//---+----//-----+
|P0000010|Lxxxxxxx|ASCII chars|
+--------+---//---+----//-----+
```

Bit-String (OPTIONAL) 43 103
16 8

primitive

This data element contains a series of bits. It uses the Qualifier data element component to record the number of bits of padding (as an 8-bit unsigned integer) needed to fill the final octet of the Data Element Contents to an even octet boundary. These padding bits have no meaning and occur in the low order bits of the final octet. The valid values for the Qualifier component are 0 through 7. The number of bits in the Data Element Contents is calculated from the following formula.

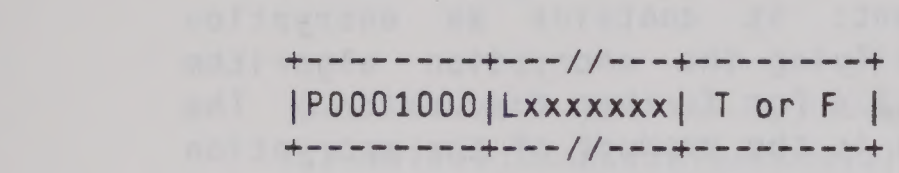
8 * number of octets in the Data Element Contents - value of Qualifier component

```
+--------+---//---+---//---+---//---+
|P1000011|Lxxxxxxx|Qxxxxxxx|  bits  |
+--------+---//---+---//---+---//---+
```

Boolean (OPTIONAL) 08 010
16 8

primitive

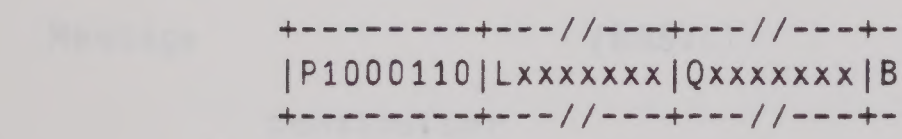
This data element contains one octet whose value is either true or false. False is represented by all bits being 0; true is represented by all bits being 1 (although any non-zero value should be interpreted as true).




Compressed (OPTIONAL) 46 106
16 8

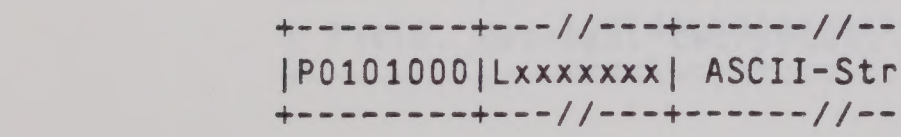
constructor

This data element must contain a Bit-String data element. It is used to represent any data that has been compressed; it may be used wherever its uncompressed contents may appear. A Qualifier data component appears in each Compressed data element; it contains a compression identifier (CID) to identify the compression algorithm used. (See Section 4.3.5.) The Data Element Contents contains the product of the compression process.




Date (BASIC) 28 050
16 constructor

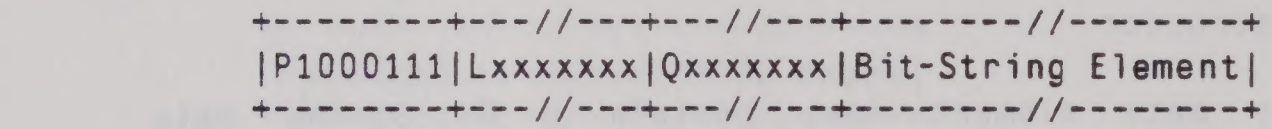
This data element contains an ASCII-String data element, which is a representation of a date and time formatted in accordance with FIPS PUBS 4 [NatB-68], 58 [NatB-79a], and 59 [NatB-79b]. The use of time and time zone is optional. It is recommended that numeric offsets be used to indicate time zone rather than alphabetic abbreviations.



Encrypted (OPTIONAL) 47 107
16 8

constructor

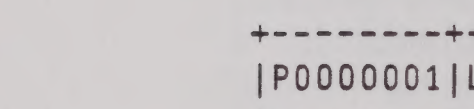
This data element must contain a Bit-String. It is used to represent any data that has been encrypted; it may be used wherever its unencrypted contents may appear. A Qualifier data component appears in each Encrypted data element; it contains an encryption identifier (EID) identifying the encryption algorithm used. (See Section 4.3.4 for further discussion.) The Data Element Contents is the product of the encryption process.




End-of-Constructor (BASIC) 01 001
16 8

primitive

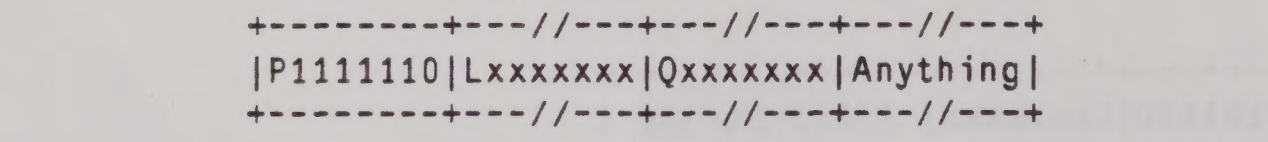
This data element terminates the Data Element Contents in a constructor data element that has indefinite length. This data element has no Contents component. (Use of this element is described in Section 4.2.2.1.)




Extension (OPTIONAL) 7E 176
16 8

either

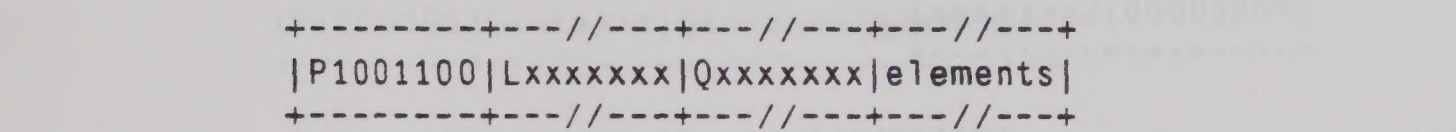
This data element is used to extend the number of available data elements beyond the 128 that are possible using a 7-bit identifier. A Qualifier component extends the encoding space for identifiers. (Extension and Vendor-Defined have the same syntax.)



Field (BASIC) 4C 114
16 8

constructor

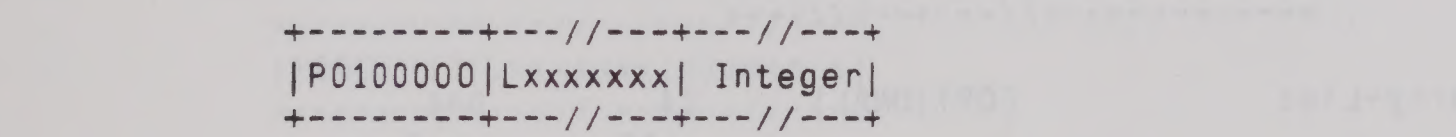
This data element uses a Qualifier data element component. The Qualifier component contains a Field Identifier (FID) indicating which specific field is being represented. (See Section 4.3.2 for further discussion.)




Integer (OPTIONAL) 20 040
16 8

primitive

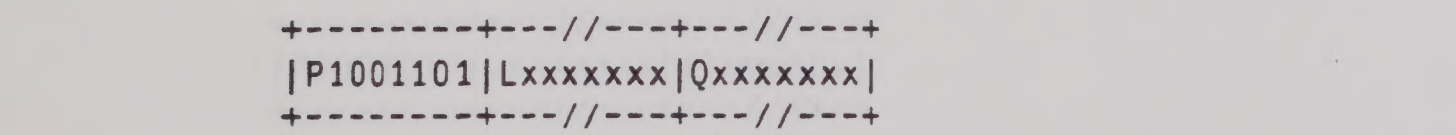
This data element contains a 2's complement integer of variable length, high-order octet first. It is recommended that the data element contents be either 2 or 4 octets long whenever possible.

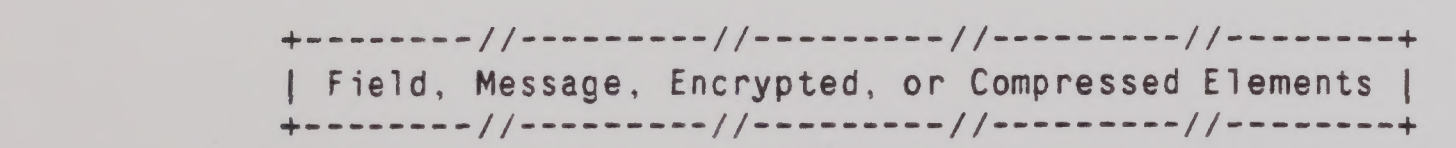



Message (BASIC) 4D 115
16 8

constructor

This data element may contain Field or Message data elements. Its Qualifier component contains a Message type (MID) indicating the type of the message. (See Section 4.3.6 for further discussion.) (The MID is completely different from the message identifier in the Message-ID field and should not be confused with it.)

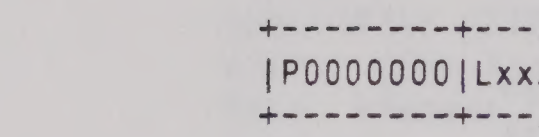


No-Op (OPTIONAL) 00 000
16 8

primitive

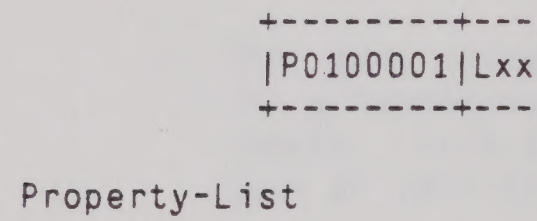
This data element does nothing. No-Op is used whenever it is necessary to include a data element that means "no operation." It is a short placeholder.




Padding (OPTIONAL) 21 041
16 8

primitive

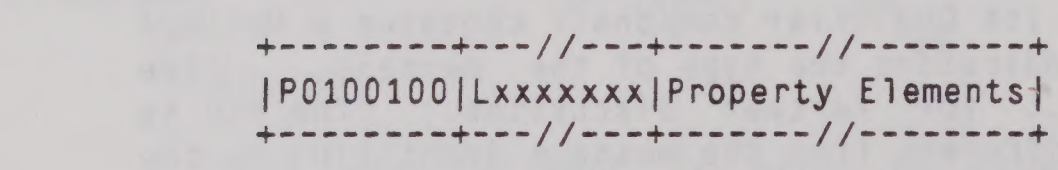
This data element is used to fill any number of octets. The contents of a Padding element are undefined and convey no information.




Property-List (OPTIONAL) 24 044
16 8

constructor

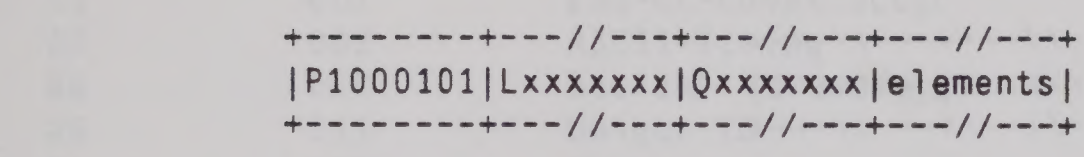
This data element contains a series of Property data elements to be associated with another data element.



Property (OPTIONAL) 45 105
16 8

constructor

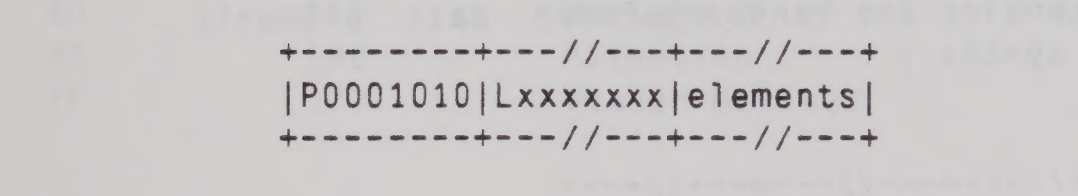
This data element uses a Qualifier data element component. The Qualifier component contains a Property-Identifier (PID) to indicate which specific property is being represented. (See Section 4.3.3 for further discussion.)




Sequence (OPTIONAL) 0A 012
16 8

constructor

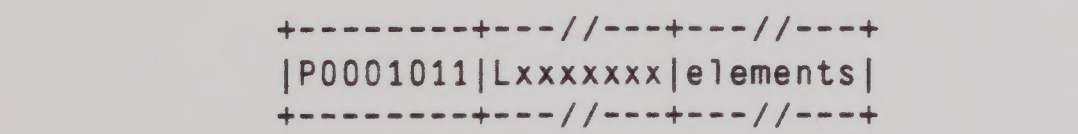
This data element contains any series of data elements. Sequence differs from Set in that the data elements making up the Data Element Contents must be considered as an ordered sequence (according to their order of appearance in the sequence.)




Set (OPTIONAL) 0B 013
16 8

constructor

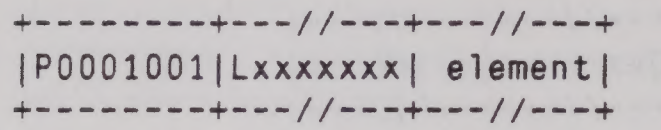
This data element contains any series of data elements with no ordering of the elements implied. (Sequence provides an ordered series.) Although the data elements contained in a Set must be stored sequentially, the order in which they are stored is not defined and not processed.



Unique-ID (OPTIONAL) 09 011
16 8

constructor

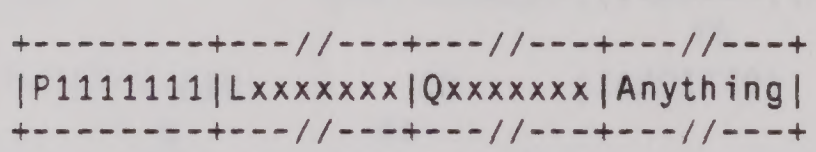
This data element is a unique identifier. It need not be human-readable. The Data Element Contents may be an ASCII-String, a Bit-String, or an Integer.




Vendor-Defined (OPTIONAL) 7F 177
16 8

either

This data element is used to represent vendor-defined data elements. A Qualifier component extends the encoding space for identifiers. The Qualifier component is not guaranteed to be unique among all interconnected systems. This data element is interpreted according to prior agreement between systems. (Extension and Vendor-Defined data elements have the same syntax.)



# APPENDIX C
# DATA ELEMENT IDENTIFIER OCTETS

| Identifier | Identifier | Data Element Name |
|---|---|---|
| 00 | 000 | No-Op |
| 01 | 001 | End-of-Constructor |
| 02 | 002 | ASCII-String |
| 08 | 010 | Boolean |
| 09 | 011 | Unique-ID |
| 0A | 012 | Sequence |
| 0B | 013 | Set |
| 20 | 040 | Integer |
| 21 | 041 | Padding |
| 24 | 044 | Property-List |
| 28 | 050 | Date |
| 43 | 103 | Bit-String |
| 45 | 105 | Property |
| 46 | 106 | Compressed |
| 47 | 107 | Encrypted |
| 4C | 114 | Field |
| 4D | 115 | Message |
| 7E | 176 | Extension |
| 7F | 177 | Vendor-Defined |

# APPENDIX D
# SUMMARY OF MESSAGE FIELDS BY COMPLIANCE CATEGORY

This appendix is for reference use. It contains no new information, but rather abstracts from that presented in Section 3.1.

This appendix contains the message field names arranged alphabetically within compliance category. (Appendix E orders the field names within functional category.) Complete field definitions appear in Appendix A.

Required fields must appear in a message. Basic fields must be recognized and processed by all CBMS systems. Optional fields need not be supported by a CBMS but, if supported, must be processed according to the meanings defined by the message format specification.

## D.1 REQUIRED Fields

From
Posted-Date
To

## D.2 BASIC Fields

Cc
Reply-To
Subject
Text

## D.3 OPTIONAL Fields

Attachments
Author
Bcc
Circulate-Next
Circulate-To
Comments

Date
End-Date
In-Reply-To
Keywords
Message-Class
Message-ID
Obsoletes
Originator-Serial-Number
Precedence
Received-Date
Received-From
References
Reissue-Type
Sender
Start-Date
Warning-Date

# APPENDIX E
# SUMMARY OF MESSAGE SEMANTICS BY FUNCTION

This appendix is for reference use. It contains no new information, but rather abstracts from that presented in Section 3.1.

This appendix contains the message field names arranged alphabetically within functional class. (Appendix D orders the field names within compliance class.) Complete field definitions appear in Appendix A.

## E.1 Circulation

Circulate-Next
Circulate-To

## E.2 Cross-Referencing

In-Reply-To
Message-ID
Obsoletes
Originator-Serial-Number
References

## E.3 Life Spans

End-Date
Start-Date
Warning-Date

## E.4 Delivery System

Received-Date
Received-From

## E.5 Miscellaneous Fields Used Generally

Attachments
Comments
Keywords
Message-Class
Precedence
Subject
Text

## E.6 Reply Generation

Reply-To

## E.7 Reissuing

Reissue-Type

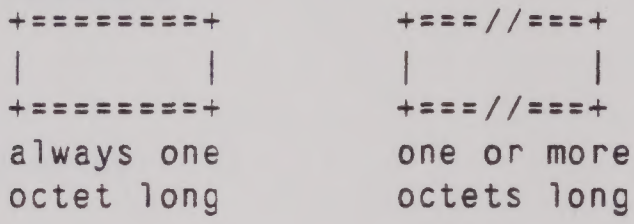
## E.8 Sending (Normal Transmission)

Author
Bcc
Cc
Date
From
Posted-Date
Sender
To

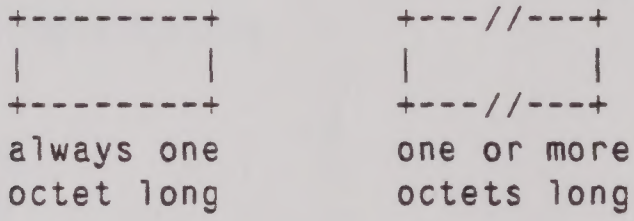
# APPENDIX F
# SUMMARY OF DATA ELEMENT SYNTAX

This appendix summarizes data element syntax by diagramming the components of data elements. Detailed presentation of data element syntax appears in Section 4.3.1.

In these diagrams, required components of a data element appear as follows. (The double border signifies "required.")




Optional components of data elements are represented as follows. (The single border signifies "not required.")



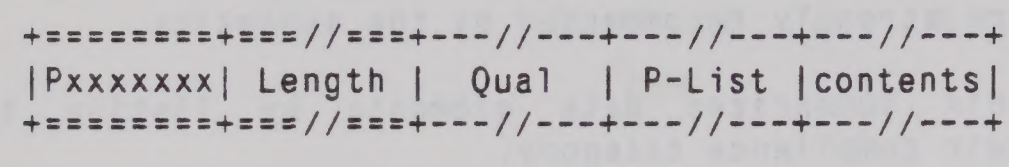

The first octet in a data element is the identifier octet. In diagrams of data elements, all eight bits of the identifier octet are always shown. Bits with fixed values show the fixed values as 1s and 0s. Bits with variable values are shown as x's and y's.

The first bit in an identifier octet is the P-bit. Its value indicates whether a data element contains a property list. (A P-bit value of 1 indicates the presence of a property list.) The remaining seven bits contain the rest of the identifier.

Other octets in a data element belong to one of four classes: Length Code, Qualifier, Property-List, and Contents. In diagrams of syntax the data element components are labeled according to their class.

| Component Class | Label |
|---|---|
| Length code | Length |
| Qualifier | Qual |
| Property-List | P-List |
| Contents | Contents |

Data elements must follow this form.




The value of the Length component is the total number of octets following the length code octet in the data element.

# APPENDIX G
# SUMMARY OF DATA ELEMENTS BY COMPLIANCE CATEGORY

Compliance categories for syntactic elements are basic and optional. Every CBMS is required to recognize and process basic elements. A CBMS is not required to process optional elements although many are strongly recommended by the semantics.

This appendix summarizes data elements by listing them according to their compliance category.

## G.1 BASIC Data Elements

| | | | |
|---|---|---|---|
| ASCII-String | (primitive) | 02<br>16 | 002<br>8 |
| Date | (constructor) | 28<br>16 | 050<br>8 |
| End-Of-Constructor | (primitive) | 01<br>16 | 001<br>8 |
| Field | (constructor) | 4C<br>16 | 114<br>8 |
| Message | (constructor) | 4D<br>16 | 115<br>8 |

## G.2 OPTIONAL Data Elements

| | | | |
|---|---|---|---|
| Bit-String | (primitive) | 43<br>16 | 103<br>8 |
| Boolean | (primitive) | 08<br>16 | 010<br>8 |
| Compressed | (constructor) | 46<br>16 | 106<br>8 |
| Encrypted | (constructor) | 47<br>16 | 107<br>8 |
| Extension | (either) | 7E<br>16 | 176<br>8 |
| Integer | (primitive) | 20<br>16 | 040<br>8 |
| No-Op | (primitive) | 00<br>16 | 000<br>8 |
| Padding | (primitive) | 21<br>16 | 041<br>8 |

| | | | |
|---|---|---|---|
| Property | (constructor) | 45 (16) | 105 (8) |
| Property-List | (constructor) | 24 (16) | 044 (8) |
| Sequence | (constructor) | 0A (16) | 012 (8) |
| Set | (constructor) | 0B (16) | 013 (8) |
| Unique-ID | (constructor) | 09 (16) | 011 (8) |
| Vendor-Defined | (either) | 7F (16) | 377 (8) |

# APPENDIX H
# EXAMPLES

This appendix presents at least one example for each of the data elements defined in this message format specification. In these examples, identifier octets are represented in binary form. All other numbers are presented in hexadecimal. ASCII strings are shown as characters rather than their numerical representation. Although this message format specification does not define the syntax of names and addresses, message originators and recipients are identified by their names. This does not imply anything about how naming and addressing can or should be done; it is simply a convenient way to identify message originators and recipients in these examples.

## H.1 Primitive Data Elements

This section contains an example of each of the primitive data elements. Each example contains a short explanation and a series of octets.

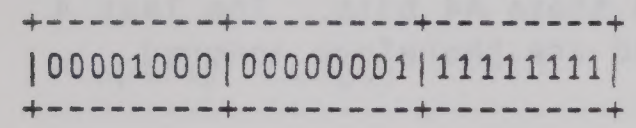
No-Op data element:

```
+--------+--------+
|00000000|00000000|
+--------+--------+
```

End-of-Constructor data element:

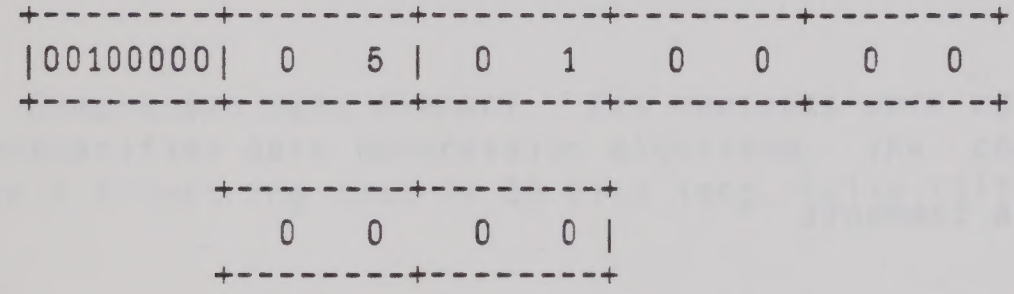
```
+--------+--------+
|00000001|00000000|
+--------+--------+
```

Boolean data element whose value is true:



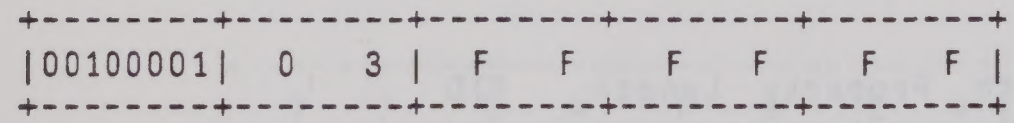
```
+--------+--------+--------+
|00001000|00000001|11111111|
+--------+--------+--------+
```

Integer data element containing five octets of data. Its value is 4,294,967,296 (decimal):



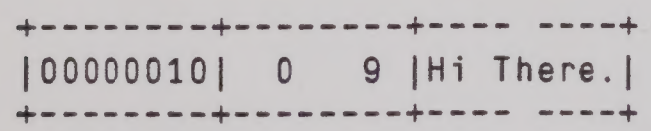
```
+--------+--------+--------+--------+--------+
|00100000|  0   5 |  0   1    0   0    0   0
+--------+--------+--------+--------+--------+

         +--------+--------+
            0   0    0   0 |
         +--------+--------+
```

Padding data element containing three octets of padding. The values of those three octets are meaningless:



```
+--------+--------+--------+--------+--------+
|00100001|  0   3 |  F   F    F   F    F   F |
+--------+--------+--------+--------+--------+
```

ASCII-String data element containing nine characters. Its value is "Hi There.":



```
+--------+--------+---- ----+
|00000010|  0   9 |Hi There.|
+--------+--------+---- ----+
```

Bit-String data element containing 44 bits of data (((7-1) x 8) - 4). Six octets are used to hold those 44 bits. The last 4 bits in the final octet are padding and are therefore ignored.

| Bit-String | Length | Spare | | |
|---|---|---|---|---|
| \|01000011\| | 0 7 \| | 0 4 \| | 0 A | 3 B |
| | 5 F | 2 9 | 1 C | D 0 \| |

## H.2 Constructor Data Elements

This section contains an example of each of the constructor data elements. Each example contains a short explanation and then an annotated series of the data elements making up the constructor.

Property-List data element containing one Property data element. The property is Printing-Name and its value is "Distribution":

| Prop-List | Length | Property | Length | PID |
|---|---|---|---|---|
| \|00100100\| | 1 1 \| | \|01000101\| | 0 F \| | 0 2 \| |

| ASCII | Length | |
|---|---|---|
| \|00000010\| | 0 C \| | \|Distribution\| |

Printing-Name Property. The value of the Printing-Name is "Distribution":

| Property | Length | PID | ASCII | Length |
|---|---|---|---|---|
| 01000101 | 0 F | 0 2 | 00000010 | 0 C |

Distribution

Compressed data element. Its contents were compressed using an unspecified data compression algorithm. The compressed data is in a bit-string that is 56 bits long, fully filling 7 octets:

| Compressed | Length | CID | Bit-String | Length |
|---|---|---|---|---|
| 01000110 | 0 B | 0 0 | 01000011 | 0 8 |

| Spare | | | |
|---|---|---|---|
| 0 0 | 1 C | 5 F | 2 D |
| 7 7 | B A | F 6 | 2 9 |

Encrypted data element. The encryption method used to encrypt its contents has been intentionally not specified. This element contains a Bit-String which contains 22 bits (((4-1) x 8) - 2) of data. These 22 bits are represented in octets; the final 2 bits in the final octet are padding and are therefore ignored:

| Encrypted | Length | EID | Bit-String | Length |
|---|---|---|---|---|
| 01000111 | 0 7 | 0 0 | 01000011 | 0 4 |

| Spare | | | | | | | |
|---|---|---|---|---|---|---|---|
| 0 | 2 | A | 3 | 7 | 8 | 1 | C |

Date data element. This example includes a date but no time. The date shown in this example is August 15, 1980:

| Date | Length | ASCII | Length | |
|---|---|---|---|---|
| 00101000 | 0 A | 00000010 | 0 8 | 19800815 |

Unique-ID data element, which is represented as an Integer data element whose value is 129 (decimal).

| Unique-ID | Length | Integer | Length | | |
|---|---|---|---|---|---|
| 00001001 | 0 4 | 00100000 | 0 2 | 0 0 | 8 1 |

Sequence data element containing two ASCII-String data elements. The first ASCII-String is "This is" while the second string is " a list":

| Sequence | Length | | ASCII | Length | | |
|---|---|---|---|---|---|---|
| 00001010 | 1 | 2 | 00000010 | 0 | 7 | This is |

| ASCII | Length | | |
|---|---|---|---|
| 00000010 | 0 | 7 | a list |

Set data element containing two Integer data elements. The first integer has a value of 519 (decimal) while the value of the second is 71 (decimal). (These two values have no ordering because they belong to a set.)

| Set | Length | | Integer | Length | | | |
|---|---|---|---|---|---|---|---|
| 00001011 | 0 | 8 | 00100000 | 0 | 2 | 0 | 2 | 0 | 7 |

| Integer | Length | | | |
|---|---|---|---|---|
| 00100000 | 0 | 2 | 0 | 0 | 4 | 7 |

Field data element. The specific field shown is the Text field with the contents "I will see you at lunch.":

| Field | Length | | FID | | ASCII | Length | |
|---|---|---|---|---|---|---|---|
| 01001100 | 1 | B | 0 | 4 | 00000010 | 1 | 8 |

| |
|---|
| I will see you at lunch. |

Message containing four fields, Posted-Date, From, Text, and To. It was sent on July 4, 1980 at 6 p.m. eastern daylight time. It is from a person named Smith. The text of the message is a question asking the recipient "Are you going to watch the fireworks?". The message is sent to Jones:

| Message | Length | Type | Field | Length |
|---|---|---|---|---|
| 01001101 | 5 A | 0 1 | 01001100 | 1 9 |

| FID | Date | Length | ASCII |
|---|---|---|---|
| 0 2 | 00101000 | 1 6 | 00000010 |

| Length | |
|---|---|
| 1 4 | 19800704-180000-0400 |

| Field | Length | FID | ASCII |
|---|---|---|---|
| 01001100 | 0 8 | 0 1 | 00000010 |

| Length | |
|---|---|
| 0 5 | Smith |

| Field | Length | FID | ASCII |
|---|---|---|---|
| 01001100 | 2 8 | 0 4 | 00000010 |

| Length |
|---|
| 2 5 |

| |
|---|
| Are you going to watch the fireworks? |

| Field | Length | FID | ASCII |
|---|---|---|---|
| 01001100 | 0 8 | 0 5 | 00000010 |

| Length | |
|---|---|
| 0 5 | Jones |

## H.3 Data Elements that Extend this Specification

This section contains examples of data elements used to extend this specification. These data elements can be either primitives or constructors, depending on the extension.

Extension data element containing a length code and 3 octets. The octet immediately following the length code identifies it as Extension Data Element 7. The Data Element Contents is the final two octets. The interpretation of the Data Element Contents would be defined in an extension or successor to this message format specification. [Note: this is an example. Any actual extension data element 7 (if it were ever used) would be completely different from anything done here.]:

| Extension | Length | | | |
|---|---|---|---|---|
| 01111110 | 0 3 | 0 7 | 4 A | E 9 |

Vendor-Defined data element containing a length code and 3 octets. The first octet identifies this as vendor-defined data element number 114 (decimal), which this particular vendor has defined to contain three printable ASCII characters in two octets. (Data element 114 (decimal) for another user would be completely different. For example, it might contain a floating point number.):

| User | Length | | | |
|---|---|---|---|---|
| 01111111 | 0 3 | 7 2 | P O | E |

## H.4 Fields

This section contains examples of Field data element constructors for each of several different fields (Keywords, Text, Subject, Vendor-Defined).

Field data element for keywords . The field contains two keywords, Message and Computer, each represented in a separate ASCII-string data element.

| Field | Length | Keywords | ASCII | Length |
|---|---|---|---|---|
| 01001100 | 1 4 | 1 4 | 00000010 | 0 7 |

| Message |
|---|

| ASCII | Length | |
|---|---|---|
| 00000010 | 0 8 | Computer |

Field data element for Text with a Property-List data element containing a comment attached. The text field contains the ASCII-String data element "Do you want lunch?"; the Property-List data element contains a comment property, which consists of an ASCII-string data element containing "Now?":

| Field | Length | Text | Prop-List | Length |
|---|---|---|---|---|
| 11001100 | 2 0 | 0 4 | 00100100 | 0 9 |

| Property | Length | PID | ASCII |
|---|---|---|---|
| 01000101 | 0 7 | 0 1 | 00000010 |

| Length | |
|---|---|
| 0 4 | Now? |

| ASCII | Length | |
|---|---|---|
| 00000010 | 1 2 | Do you want lunch? |

Field data element for Subject containing an ASCII-String data element ("Good restaurants in Detroit" followed by a carriage return and a line feed). (A recipient would expect the message to contain some information about restaurants in the Detroit area.):

| Field | Length | Subject | ASCII | Length |
|---|---|---|---|---|
| 01001100 | 2 1 | 0 7 | 00000010 | 1 E |

| |
|---|
| Good restaurants in Detroit.<cr><lf> |

Field data element whose form and meaning was defined by a vendor. This vendor has defined vendor-defined field 12 (decimal) to be a field with a printing name of "Reply-by" and contents consisting of a date; January 7, 1981 in this case. (The meaning of vendor-defined field 12 is unique to the vendor; the same field number would have different meaning for other vendors.):

| Field | Length | Qualifier | User | number |
|---|---|---|---|---|
| 11001100 | 1 F | 8 2 | 0 0 | 0 C |

| Prop-List | Length | Property | Length |
|---|---|---|---|
| 00100100 | 0 E | 01000101 | 0 C |

| PID | ASCII | Length | |
|---|---|---|---|
| 0 2 | 00000010 | 0 9 | Reply-By: |

| Date | Length | ASCII | Length |
|---|---|---|---|
| 00101000 | 0 A | 00000010 | 0 8 |

| |
|---|
| 19810107 |

## H.5 Messages

This section contains several examples of complete messages and shows the results of reissuing a message. (See Section 3.2.2.)

The following sample message had Stevens as its originator and Johnson as its recipient. The message was sent on August 14, 1980 at 10 a.m. EDT. The subject of the message is "Project Deadline" and the message is a reminder that the deadline is the next day and that the section of the report for the project being done by Johnson should be turned in to Stevens by 3 p.m. that day.

| Message | Length | | Type | |
|---|---|---|---|---|
| 01001101 | 8 | 1 | B | 6 | 0 | 1 |

| Field | Length | | FID | ASCII |
|---|---|---|---|---|
| 01001100 | 0 | A | 0 | 5 | 00000010 |

| Length | |
|---|---|
| 0 | 7 |

| Johnson |
|---|

| Field | Length | | FID | ASCII |
|---|---|---|---|---|
| 01001100 | 0 | A | 0 | 1 | 00000010 |

| Length | |
|---|---|
| 0 | 7 |

| Stevens |
|---|

| Field | Length | | FID | ASCII | Length | |
|---|---|---|---|---|---|---|
| 01001100 | 1 | 3 | 0 | 7 | 00000010 | 1 | 0 |

| Project Deadline |
|---|

| Field | Length | | FID | Date | Length | |
|---|---|---|---|---|---|---|
| 01001100 | 1 | 7 | 0 | 2 | 00101000 | 1 | 4 |

```
  ASCII    Length
+--------+--------+----                ----+
|00000010|  1   2 |19800814-1000-0400|
+--------+--------+----                ----+

  Field    Length    FID      ASCII   Length
+--------+--------+--------+--------+--------+
|01001100|  6   D |  0   4 |00000010|  6   A |
+--------+--------+--------+--------+--------+

        +----
        |Don't forget the project report is
        +----

         due tomorrow.  Please have<CrLf>

         your section to me by three this

              -----+
         afternoon.|
              -----+
```

The following example illustrates the results of reissuing the first message in this section. This message contains the original message (as a Message data element), To, From, and Posted-Date fields, and a Reissue-Type field with Redistributed as its value:

```
 Message       Length        Type
+--------+--------+--------+--------+
|01001101|  8   1 |  F   C |  0   1 |
+--------+--------+--------+--------+

   Field   Length    FID      ASCII
+--------+--------+--------+--------+
|01001100|  0   9 |  0   5 |00000010|
+--------+--------+--------+--------+

          Length
        +--------+--  --+
        |  0   6 |Cooper|
        +--------+--  --+
```

| Field | Length | | FID | | ASCII |
|---|---|---|---|---|---|
| 01001100 | 0 | A | 0 | 1 | 00000010 |

| Length | |
|---|---|
| 0 | 7 |

| Johnson |
|---|

| Field | Length | | FID | | Date | Length | |
|---|---|---|---|---|---|---|---|
| 01001100 | 1 | 7 | 0 | 2 | 00101000 | 1 | 4 |

| ASCII | Length | |
|---|---|---|
| 00000010 | 1 | 2 |

| 19800814-1030-0400 |
|---|

| Field | Length | | FID | | ASCII | Length | |
|---|---|---|---|---|---|---|---|
| 01001100 | 1 | 0 | 2 | 5 | 00000010 | 0 | D |

| Redistributed |
|---|

| Message | Length | | Type | |
|---|---|---|---|---|
| 01001101 | 8 | 1 | B | 6 | 0 | 1 |

| Field | Length | | FID | | ASCII |
|---|---|---|---|---|---|
| 01001100 | 0 | A | 0 | 5 | 00000010 |

| Length | |
|---|---|
| 0 | 7 |

| Johnson |
|---|

| Field | Length | | FID | | ASCII |
|---|---|---|---|---|---|
| 01001100 | 0 | A | 0 | 1 | 00000010 |

```
  Length
 +--------+--- ---+
 |  0   7 |Stevens|
 +--------+--- ---+
```

| Field | Length | | FID | | ASCII | Length | |
|---|---|---|---|---|---|---|---|
| 01001100 | 1 | 3 | 0 | 7 | 00000010 | 1 | 0 |

```
 +----          ----+
 |Project Deadline|
 +----          ----+
```

| Field | Length | | FID | | Date | Length | |
|---|---|---|---|---|---|---|---|
| 01001100 | 1 | 7 | 0 | 2 | 00101000 | 1 | 4 |

```
  ASCII    Length
 +--------+--------+----                 ----+
 |00000010|  1   2 |19800814-1000-0400|
 +--------+--------+----                 ----+
```

| Field | Length | | FID | | ASCII | Length | |
|---|---|---|---|---|---|---|---|
| 01001100 | 6 | D | 0 | 4 | 00000010 | 6 | A |

```
 +----
 |Don't forget the project report is
 +----

  due tomorrow.  Please have<CrLf>

  your section to me by three this

        ----+
  afternoon.|
        ----+
```

## H.6 Unknown Lengths

This section contains two examples of data elements with an unknown length. The two examples have been presented in sections H.2 and H.5, but with a known rather than an unknown length.

Set data element with an unknown length containing two Integer data elements. The first integer has a value of 519 (decimal) while the value of the second is 71 (decimal). (These two values have no ordering because they belong to a set.)

| Set | Length | Integer | Length | | |
|---|---|---|---|---|---|
| 00001011 | 8 0 | 00100000 | 0 2 | 0 2 | 0 7 |

| Integer | Length | | |
|---|---|---|---|
| 00100000 | 0 2 | 0 0 | 4 7 |

| End-of-Con | Length |
|---|---|
| 00000000 | 00000000 |

The following sample message with an unknown length had Stevens as its originator and Johnson as its recipient. The message was sent on August 14, 1980 at 10 a.m. EDT. The subject of the message is "Project Deadline" and the message is a reminder that the deadline is the next day and that the section of the report for the project being done by Johnson should be turned in to Stevens by 3 p.m. that day.

| Message | Length | Type |
|---|---|---|
| 01001101 | 8 0 | 0 1 |

| Field | Length | FID | ASCII |
|---|---|---|---|
| 01001100 | 0 A | 0 5 | 00000010 |

| Length | |
|---|---|
| 0 7 | Johnson |

| Field | Length | | FID | | ASCII |
|---|---|---|---|---|---|
| 01001100 | 0 | A | 0 | 1 | 00000010 |

| Length | |
|---|---|
| 0 | 7 |

| Stevens |
|---|

| Field | Length | | FID | | ASCII | Length | |
|---|---|---|---|---|---|---|---|
| 01001100 | 1 | 3 | 0 | 7 | 00000010 | 1 | 0 |

| Project Deadline |
|---|

| Field | Length | | FID | | Date | Length | |
|---|---|---|---|---|---|---|---|
| 01001100 | 1 | 7 | 0 | 2 | 00101000 | 1 | 4 |

| ASCII | Length | | |
|---|---|---|---|
| 00000010 | 1 | 2 | 19800814-1000-0400 |

| Field | Length | | FID | | ASCII | Length | |
|---|---|---|---|---|---|---|---|
| 01001100 | 6 | D | 0 | 4 | 00000010 | 6 | A |

| Don't forget the project report is |
|---|

due tomorrow. Please have<CrLf>

your section to me by three this

afternoon.|

| End-of-Con | Length |
|---|---|
| 00000000 | 00000000 |

## H.7 Message Encoding Using Vendor-Defined Fields

This example is provided to illustrate the encoding of non-FIPS format messages in the FIPS format. It is the intent of the standard to deal with computer based message systems which are primarily intended for person-to-person use. This example deals with the definition and use of vendor-defined fields to extend the use of the standard to station-to-station messaging. The context is a military message using the military standard JANAP-128 format.

### H.7.1 Example of a JANAP-128 Message

```
JANAP-128
RTTUZYUW RUABCDE0010 0330930-UUUU--RUXABYE.
ZNR UUUUU
R 020830Z FEB 82
FM Commander,Atlantic Fleet
TO USS SHIPA
BT
UNCLAS

 MESSAGE        BODY

BT
#0010
NNNN
```

### H.7.2 Encoding of Example using the FIPS Message Format

| Message | Length | | Type | |
|---|---|---|---|---|
| 01001101 | 8 | 1 | D | 0 | 0 | 1 |

| Field | Length | | FID | |
|---|---|---|---|---|
| 01001100 | 0 | 4 | 1 | 8 |

| ASCII | Length | |
|---|---|---|
| 00000010 | 0 1 | R |

| Field | Length | FID | | |
|---|---|---|---|---|
| 01001100 | 0 7 | 8 2 | 0 0 | 0 1 |

| ASCII | Length | | |
|---|---|---|---|
| 00000010 | 0 2 | T | T |

| Field | Length | FID | | |
|---|---|---|---|---|
| 01001100 | 0 6 | 8 2 | 0 0 | 0 2 |

| ASCII | Length | |
|---|---|---|
| 00000010 | 0 1 | U |

| Field | Length | FID | | |
|---|---|---|---|---|
| 01001100 | 0 9 | 8 2 | 0 0 | 0 3 |

| ASCII | Length | |
|---|---|---|
| 00000010 | 0 4 | ZYUW |

| Field | Length | FID |
|---|---|---|
| 01001100 | 0 A | 2 2 |

| ASCII | Length | |
|---|---|---|
| 00000010 | 0 7 | RUABCDE |

| Field | Length | FID |
|---|---|---|
| 01001100 | 0 7 | 1 7 |

| ASCII | Length | | |
|---|---|---|---|
| 00000010 | 0 4 | 0010 | |

| Field | Length | FID | Date | Length |
|---|---|---|---|---|
| 01001100 | 1 8 | 0 2 | 00101000 | 1 5 |

| ASCII | Length | | |
|---|---|---|---|
| 00000010 | 1 3 | 19820202093000-0000 | |

| Field | Length | FID | | |
|---|---|---|---|---|
| 01001100 | 0 9 | 8 2 | 0 0 | 0 2 |

| ASCII | Length | | |
|---|---|---|---|
| 00000010 | 0 4 | UUUU | |

| Field | Length | FID | | |
|---|---|---|---|---|
| 01001100 | 0 C | 8 2 | 0 0 | 0 4 |

| ASCII | Length | | |
|---|---|---|---|
| 00000010 | 0 7 | RUXABYE | |

| Field | Length | FID | | |
|---|---|---|---|---|
| 01001100 | 0 A | 8 2 | 0 0 | 0 2 |

| ASCII | Length | | |
|---|---|---|---|
| 00000010 | 0 5 | UUUUU | |

| Field | Length | FID |
|---|---|---|
| 01001100 | 0 4 | 1 8 |

| ASCII | Length | |
|---|---|---|
| 00000010 | 0 1 | R |

| Field | Length | FID | Date | Length |
|---|---|---|---|---|
| 01001100 | 1 4 | 1 1 | 00101000 | 1 1 |

| ASCII | Length | | |
|---|---|---|---|
| 00000010 | 0 F | 8202020830-0000 | |

| Field | Length | FID |
|---|---|---|
| 01001100 | 1 B | 0 1 |

| ASCII | Length | |
|---|---|---|
| 00000010 | 1 8 | |

| | | |
|---|---|---|
| | | Commander,Atlantic Fleet |

| Field | Length | FID |
|---|---|---|
| 01001100 | 0 C | 0 5 |

| ASCII | Length | | |
|---|---|---|---|
| 00000010 | 0 9 | USS SHIPA | |

| Field | Length | FID |
|---|---|---|
| 01001100 | 0 7 | 0 4 |

| ASCII | Length | | |
|---|---|---|---|
| 00000010 | 0 4 | BODY | |

| Field | Length | FID |
|---|---|---|
| 01001100 | 0 7 | 1 7 |

| ASCII | Length | |
|---|---|---|
| 00000010 | 0 4 | 0010 |

H.7.3 Field Mappings of JANAP-128 to FIPS Format

| JANAP-128 Field | FIPS Format Field |
|---|---|
| Precedence | Precedence (Appendix A) |
| Language Media Format | Vendor-Defined |
| Security | Vendor-Defined |
| Content Indicator Code | Vendor-Defined |
| Origination Station Routing Indication | Sender (Appendix A) |
| Station Serial Number | Originator-Serial-Number (Appendix A) |
| Time of File | Posted-Date (Appendix A) |
| Security | Vendor-Defined |
| Destination Station Routing Indicator | Vendor-Defined |
| Security | Vendor-Defined |
| Precedence | Precedence (Appendix A) |
| Date/Time Group | Date (Appendix A) |
| FM | From (Appendix A) |
| TO | To (Appendix A) |
| Body of Message | Text (Appendix A) |
| Station Serial Number | Originator-Serial-Number (Appendix A) |

H.7.4 Vendor-Defined Fields

| Field Name | Identifier Value |
|---|---|
| | 8 |
| Description | |

Language Media Format 01
8

This field contains two ASCII characters; the first indicates the input media and the second the output media.

Security 02
8

This field contains a variable length ASCII character indicator of the security classification of the messages.

Content Indicator Code 03
8

This field contains four ASCII characters and provides information describing the message content and message handling actions to be performed.

Destination Station Routing Indicator 04
8

This field contains four ASCII characters indicating the CPU and terminal device to which the message should be sent.

# REFERENCES


[BlaR-80]
R. P. Blanc and J. F. Heafner. The NBS Program in Computer Network Protocol Standards. In Proceedings, ICCC 80. 1980.

[CCIT-82]
CCITT Study Group VII/5. Draft recommendation X.MHS1: Message Handling Systems: System Model - Service Elements (Version 2). Technical Report, International Telegraph and Telephone Consultative Committee (CCITT), December, 1982.

[CroD-77]
David H. Crocker, John J. Vittal, Kenneth T. Pogran, D. Austin Henderson, Jr. Standard for the Format of ARPA Network Text Messages. RFC 733, The Rand Corporation, Bolt Beranek and Newman Inc, Massachussets Institute of Technology, Bolt Beranek and Newman Inc., November, 1977.

[FeiE-79]
E. Feinler, J. Pickens, and A. Sjoberg. Computer Message Services Bibliography. Technical Report NIC-BIBLIO-791201, SRI International, December, 1979.

[ISOD-79]
ISO/TC97/SC6 Data Communications. Second Draft Proposed Communication Heading Format Standard. ISO/TC97/SC6 N 1948, ISO International Organization for Standardization Organization Internationale de Normalisation, September, 1979. Secretariat: USA (ANSI).

[ISOD-82]
ISO/TC97/SC16. Information Processing Systems - Open Systems Interconnection - Basic Reference Model. ISO/DIS 7498, ISO International Organization for Standardization Organization Internationale de Normalisation, December, 1982.

[NatB-68]
National Bureau of Standards. Calendar Date. Federal Information Processing Standards Publication 4, U.S. Department of Commerce / National Bureau of Standards, November, 1968.

[NatB-75]
National Bureau of Standards. Code Extension Techniques in 7 or 8 Bits. Federal Information Processing Standards Publication 35, U.S. Department of Commerce / National Bureau of Standards, June, 1975.

[NatB-77]
National Bureau of Standards. Data Encryption Standard. Federal Information Processing Standards Publication 46, U.S. Department of Commerce / National Bureau of Standards, January, 1977.

[NatB-79a]
National Bureau of Standards. Representations of Local Time of the Day for Information Interchange. Federal Information Processing Standards Publication 58, U.S. Department of Commerce / National Bureau of Standards, February, 1979.

[NatB-79b]
National Bureau of Standards. Representations of Universal Time, Local Time Differentials, and United States Time Zone References for Information Interchange. Federal Information Processing Standards Publication 59, U.S. Department of Commerce / National Bureau of Standards, February, 1979.

[NatB-80]
National Bureau of Standards. Code for Information Interchange. Federal Information Processing Standards Publication 1-1, U.S. Department of Commerce / National Bureau of Standards, December, 1980.

[PosJ-79]
Jonathan B. Postel. INTERNET MESSAGE PROTOCOL. RFC 753, Information Sciences Institute, March, 1979.

[TasG-80]
Task Group X3S33 on Data Communications Formats, ANSI Subcommittee X3S3 on Data Communications. Third Draft Proposed American National Standard for Heading Format Structure for Code Independent Communication Headings. ANSI document X3S37/80-01, Computer and Business Equipment Manufacturers Association, 1980.

# INDEX


ASCII-String  35, 36, 47, 50, 52, 54, 58, 59, 60, 61, 63, 65, 69
Assignment  22, 28, 61
Attachments  23, 57
Author  19, 57

BASIC  18
BASIC Data Elements
  ASCII-String  47, 63
  Date  50, 65
  End-of-Constructor  48, 66
  Field  50, 66
  Message  51, 67
BASIC fields
  Cc  20
  Reply-To  19
  Subject  23
  Text  23
BASIC syntactic elements  35
Bcc  19, 25, 57
Bit numbering in octets  37
Bit-String  36, 42, 47, 49, 50, 52, 64, 65, 69
Boolean  36, 48, 64

Cc  20, 58
Chains of correspondence  29
Circulate-Next  20, 31, 58
Circulate-To  20, 31, 58
Circulation  31
Comment  36, 37, 44, 54
Comments  23, 58
Compliance requirements  41
Compressed  37, 43, 49, 54, 65
Compression identifier  49, 65
Compression Identifiers
  Unspecified  54
Constructor data element  35, 36
Contents  38, 76
Cross Referencing  29

Data Element Contents  43, 44, 87, 42, 44, 52, 69, 42, 44, 46, 47, 51, 64, 69, 87
Data Elements
  ASCII-String (BASIC)  47, 63
  Bit-String (OPTIONAL)  47, 64

Boolean (OPTIONAL) 48, 64
Compressed (OPTIONAL) 49, 65
Date (BASIC) 50, 65
Encrypted (OPTIONAL) 50, 65
End-of-Constructor (BASIC) 48, 66
Extension (OPTIONAL) 52, 66
Field (BASIC) 50, 66
Integer (OPTIONAL) 48, 67
Message (BASIC) 51, 67
No-Op (OPTIONAL) 49, 67
Padding (OPTIONAL) 49, 68
Property (OPTIONAL) 51, 68
Property-List (OPTIONAL) 51, 68
Sequence (OPTIONAL) 51, 69
Set (OPTIONAL) 52, 69
Unique-ID (OPTIONAL) 52, 69
Vendor-Defined (OPTIONAL) 53, 70
Date 20, 50, 58, 60, 61, 62, 65
Dating 30
Delivery 13, 20, 60
Delivery Protocol 13
Delivery Slot 13

Encapsulating 26
Encrypted 37, 43, 50, 54, 65
Encryption identifier 50, 65
Encryption Identifiers
FIPS-Standard 54
Unspecified 54
End-Date 20, 21, 30, 58, 62
End-Of-Constructor 36, 42, 44, 48, 66
Extension 35, 46, 52, 66

Field 14, 31, 35, 36, 37, 43, 50, 51, 66, 67, 72
Field Identifier 50, 66
Field label presentation 35
Fields
Attachments (OPTIONAL) 57, 23
Author (OPTIONAL) 57, 19
Bcc (OPTIONAL) 57, 19
Cc (BASIC) 58, 20
Circulate-Next (OPTIONAL) 58, 20
Circulate-To (OPTIONAL) 58, 20
Comments (OPTIONAL) 58, 23
Date (OPTIONAL) 58, 20
End-Date (OPTIONAL) 58, 20
From (REQUIRED) 59, 19
In-Reply-To (OPTIONAL) 59, 21
Keywords (OPTIONAL) 59, 23
Message-Class (OPTIONAL) 59, 22
Message-ID (OPTIONAL) 59, 21

Obsoletes (OPTIONAL) 59, 21
Originator-Serial-Number (OPTIONAL) 59, 21
Posted-Date (REQUIRED) 60, 20
Precedence (OPTIONAL) 60, 22
Received-Date (OPTIONAL) 60, 20
Received-From (OPTIONAL) 60, 22
References (OPTIONAL) 60, 22
Reissue-Type (OPTIONAL) 61, 22
Reply-To (BASIC) 61, 19
Sender (OPTIONAL) 61, 19
Start-Date (OPTIONAL) 61, 21
Subject (BASIC) 61, 23
Text (BASIC) 61, 23
To (REQUIRED) 61, 19
Warning-Date (OPTIONAL) 62, 21
FIPS-Standard 54, 55
From 17, 19, 29, 57, 59, 61

Globally unique identifiers 29

Identifier octet 38, 41, 37, 38, 42, 44, 76
Identifiers
globally unique 29
In-Reply-To 21, 29, 59
Indefinite length code 41
Integer 36, 48, 52, 67, 69

Keywords 23, 59, 88

Length Code 40, 41, 42, 38, 39, 40, 41, 42, 44, 46, 76, 77, 87
Long length code 41

Message Transfer System 13, 22, 60
Message 14, 17, 35, 36, 37, 43, 51, 67
Message content 13
Message envelope 13
Message stores 30
Message Transfer System 13, 22, 60, 12, 13, 14, 17, 20, 22, 60
Message Types
FIPS-Standard 55
Message-Class 22, 59
Message-ID 21, 22, 29, 31, 59, 60

No-Op 49, 67
Numbering bits in octets 37

Obsoletes 21, 29, 59
Octets
bit numbering in 37

OPTIONAL 18
OPTIONAL Data Elements
Bit-String 47, 64
Boolean 48, 64
Compressed 49, 65
Encrypted 50, 65
Extension 52, 66
Integer 48, 67
No-Op 49, 67
Padding 49, 68
Property 51, 68
Property-List 51, 68
Sequence 51, 69
Set 52, 69
Unique-ID 52, 69
Vendor-Defined 53, 70
OPTIONAL fields
Attachments 23
Author 19
Bcc 19
Circulate-Next 20
Circulate-To 20
Comments 23
Date 20
End-Date 20
In-Reply-To 21
Keywords 23
Message-Class 22
Message-ID 21
Obsoletes 21
Originator-Serial-Number 21
Precedence 22
Received-Date 20
Received-From 22
References 22
Reissue-Type 22
Sender 19
Start-Date 21
Warning-Date 21
OPTIONAL syntactic elements 35
Originator 15, 18, 20, 30, 57, 58, 59, 61
Originator-Serial-Number 21, 30, 59

Padding 49, 68
Person 18
Posted-Date 17, 20, 31, 58, 60
Posting 13
Posting Protocol 13
Posting Slot 13
Precedence 22, 60
Precedence categories 22

Precedence scheme 60
Presentation
  field label 35
Primitive data element 36, 35, 36
Printing-Name 36, 37, 44, 54, 82
Process 18
Properties
  Comment 54
  Printing-Name 54
Property 38, 43, 51, 68
Property-Identifier 51, 68
Property-List 36, 38, 44, 46, 51, 68, 76

Qualifier 38, 39, 40, 41, 42, 43, 44, 46, 47, 49, 50,
    51, 52, 53, 64, 65, 66, 68, 70, 76
Qualifiers 43

Received-Date 20, 60
Received-From 22, 60
Recipient 15, 19, 20, 22, 57, 58, 60, 61
Redistribution 22, 26, 61
References 22, 29, 60
Reissue-Type 22, 61
Reply 18, 28
Reply-to 19, 28, 59, 61
REQUIRED 18
REQUIRED fields
  From 19
  Posted-Date 20
  To 19
Requirements
  compliance 41
Role 18

Sender 19, 31, 61
Sequence 35, 36, 51, 69
Sequences 36
Serial Numbers 21, 30, 59
Set 36, 51, 52, 69
Short length code 41
Slot 13
Start-Date 21, 30, 61
Subject 23, 61
Syntactic reissuing 26

Text 23, 32, 61
To 17, 19, 31, 37, 61

Unique identifiers 29
Unique-ID 52, 59, 60, 69
Unspecified 54

User Agent 12, 13, 14
User interface 35

Vendor-Defined 35, 46, 53, 70

Warning-Date 21, 30, 62